Pinocchio Effect in AI Hallucinations

Kishore Chalakkal Varghese, PhD.

2026

Pinocchio Effect in AI Hallucinations

First edition: 2026

Note on the use of AI. — Artificial intelligence tools (Perplexity and ChatGPT) were used exclusively for formal checks, minor corrections, and typographical consistency review. All content, interpretations, and textual choices remain entirely attributable to the author.

Independently printed via
Amazon Kindle Direct Publishing

to my family

Contents

List of Figures

List of Tables

Preface

Artificial intelligence has learned to speak with remarkable fluency.

It can explain scientific theories, summarize legal documents, generate computer code, compose business reports, translate between languages, and sustain conversations that appear coherent, informed, and purposeful. Its sentences are often well structured. Its tone may be calm, authoritative, and precise. It can sound like a teacher, an engineer, a physician, a consultant, or a trusted colleague.

And yet, at times, it invents.

It may cite an article that was never published, attribute words to a person who never spoke them, describe an event that did not occur, combine unrelated facts, or present an uncertain inference as established knowledge. The answer may be false while remaining grammatically perfect, logically arranged, and stylistically persuasive.

This is the central tension explored in this book.

The phrase *Pinocchio Effect* is used here as a metaphor, but with an important limitation. Pinocchio lies knowingly. His growing nose reveals an intentional departure from the truth. A large language model, by contrast, does not necessarily possess intention, self-awareness, or an internal understanding that a statement is false. It does not need to decide to deceive before producing an inaccurate answer.

The comparison therefore does not suggest that artificial intelligence lies in the same way a human being lies.

It describes something more subtle: the human perception of intentional knowledge behind machine-generated language. When an artificial system speaks with confidence, users naturally interpret that confidence

as evidence of understanding. When the system is wrong, the error can feel like deception because the form of the answer resembles the form of informed testimony.

The machine does not grow a visible nose. Its error is concealed inside the fluency of its language.

This book examines how that concealment becomes possible.

The problem of AI hallucination is often described as though it were a temporary defect that will disappear when models become sufficiently large or powerful. Progress in model architecture, training data, retrieval systems, verification techniques, and tool use has undoubtedly reduced many forms of error. However, hallucination is not simply an accidental imperfection attached to language models from the outside.

It is connected to the way these systems are built.

A large language model is trained to predict plausible continuations of sequences. It learns statistical regularities from enormous collections of text and encodes these regularities in distributed numerical representations. This process gives the model extraordinary generative power. It also creates a fundamental separation between linguistic probability and factual truth.

The most probable continuation is not always the correct continuation.

A sentence can be grammatically valid, semantically coherent, and factually false. A bibliographic reference can follow every convention of academic writing and still refer to a nonexistent publication. A technical explanation can contain correct terminology while reaching an unsupported conclusion. A response can be internally consistent yet disconnected from external evidence.

This distinction is easy to state but difficult to preserve in practice.

Human beings are deeply influenced by form. We associate clarity with competence, precision with knowledge, and confidence with authority. These associations are useful in ordinary communication, but they become dangerous when applied automatically to generative systems. Language models can reproduce the external signs of expertise without

possessing the same relationship to evidence, accountability, experience, or uncertainty as a human expert.

The risk therefore does not arise from the model alone.

It emerges from the interaction between machine generation and human interpretation.

An AI hallucination becomes consequential when a person accepts it, acts on it, shares it, or embeds it in a decision-making process. The same false answer may be harmless in a fictional exercise and dangerous in medicine, law, finance, engineering, cybersecurity, or public administration. Risk depends not only on whether a statement is false, but also on the context in which it is used, the authority attributed to the system, the difficulty of verification, and the consequences of error.

For this reason, the study of hallucination cannot remain only a technical discussion about model accuracy.

It is also a question of system design, organizational governance, human psychology, professional responsibility, and epistemology. It asks how we know that a machine-generated statement is supported. It asks what evidence should accompany an answer. It asks when a system should abstain, when a human should intervene, and who remains accountable when an automated output causes harm.

This book follows that broader path.

The first chapters explain how large language models work, what an AI hallucination is, and why such hallucinations occur. The discussion then moves from causes to mitigation, detection, measurement, and risk control. The final part examines the Pinocchio Effect itself: why people trust false machine outputs, how interface design and conversational fluency influence judgment, and how trust can be made more proportionate to evidence.

The technical sections include mathematical formulations, architectural concepts, evaluation metrics, retrieval mechanisms, and system-level controls. They are intended to provide precision rather than unnecessary complexity. Readers do not need to be specialists in machine learning, but they should be willing to look beneath the visible surface of a

generated answer.

The purpose is not to portray artificial intelligence as inherently deceptive or untrustworthy.

Such a conclusion would be both simplistic and inaccurate. Large language models are valuable instruments. They can extend human capability, accelerate analysis, support creativity, improve access to information, and assist in complex professional tasks. Their limitations do not cancel their usefulness.

But usefulness without understanding can become dependence, and dependence without verification can become risk.

The appropriate response is therefore neither blind enthusiasm nor instinctive rejection. It is disciplined trust.

Disciplined trust requires us to distinguish between fluency and evidence, between generation and retrieval, between confidence and calibration, and between assistance and authority. It requires systems that expose uncertainty rather than hide it, organizations that define responsibility rather than diffuse it, and users who understand that a convincing answer is not the same as a verified answer.

The deeper issue is not whether machines will eventually stop making mistakes. No complex information system can be expected to eliminate error completely.

The more important question is whether we can design technical and human environments in which errors are visible, bounded, detectable, and correctable before they become decisions.

Pinocchio's nose made falsehood physically apparent.

Artificial intelligence offers no such natural warning.

The responsibility for creating that warning now belongs to us.

1

How Large Language Models Work

A large language model does not begin with truth and then choose words. It begins with words—or, more precisely, tokens—and constructs a probable continuation. The distinction is the technical doorway through which hallucinations enter.

1.1 Why the Mechanism Matters

Large Language Models (LLMs) are often described through their visible abilities: they answer questions, write code, summarize reports, translate languages, draft legal clauses, explain scientific concepts, and sustain apparently coherent conversations. These capabilities are real. Yet they can also obscure the simpler computational objective at the core of most modern generative language systems.

An autoregressive LLM is trained to estimate the probability of the next token given the tokens that came before it. Its basic operation can be written as

$$P(x_t \mid x_1, x_2, \ldots, x_{t-1}), \tag{1.1}$$

where x_t is the next token and $x_1, \ldots, x_{t-1}$ form the available context. A complete sequence is modelled through the chain rule of probability:

$$P(x_1, x_2, \ldots, x_n) = \prod_{t=1}^{n} P(x_t \mid x_{<t}), \tag{1.2}$$

where $x_{<t}$ denotes all tokens preceding position t.

This objective appears narrow, but it exerts broad pressure on the model. To predict the next token well across billions or trillions of examples, the model must internalize many regularities of human-generated data: syntax, semantics, discourse structure, factual associations, source conventions, programming patterns, argument forms, and stylistic cues.

The result is a system that can behave as though it possesses a large body of knowledge. However, its generation objective does not directly require it to verify whether a statement corresponds to reality. The model is optimized to produce a probable continuation, not to perform an independent truth test on every proposition.

Key idea

The central technical distinction of this book is the difference between **linguistic probability** and **epistemic validity**. An output can be highly probable under the model and still be false in the world.

1.2 From Text to Tokens

An LLM does not ordinarily process words as indivisible linguistic units. Before computation begins, a tokenizer converts text into a sequence of discrete symbols called *tokens*. A token may represent a complete word, a subword, punctuation, whitespace, a byte sequence, or a fragment of a number.

For example, depending on the tokenizer, the word

`unbelievable`

might be segmented approximately as

`un` + `believ` + `able`.

Subword tokenization provides a practical compromise. Word-level vocabularies become extremely large and handle rare words poorly, while character-level representations produce long sequences. Subwords allow frequent units to remain compact while unfamiliar words can still be decomposed into known pieces.

Common tokenization families include Byte Pair Encoding (BPE), WordPiece, SentencePiece, unigram language-model tokenization, and byte-level variants. The tokenizer defines a vocabulary of size V and maps each token to an integer identifier:

$$\text{text} \longrightarrow (i_1, i_2, \ldots, i_n), \qquad i_k \in \{1, \ldots, V\}. \tag{1.3}$$

Tokenization is not a neutral preprocessing detail. It influences sequence length, multilingual efficiency, number handling, code representation, and the cost of inference. It can also contribute to characteristic errors. A model may struggle with exact spelling, arithmetic strings, unusual names, or rare domain terms partly because the underlying token boundaries are inconvenient.

Technical note: tokens are not words

The phrase "Large language models reason" may contain more or fewer tokens than its visible word count. Token boundaries depend on the exact tokenizer. Consequently, a context window advertised as, for example, 128,000 tokens does not correspond to a fixed number of pages or words.

1.3 Embeddings: Turning Symbols into Vectors

Token identifiers are categorical labels; the integer assigned to a token has no intrinsic semantic meaning. The model therefore converts each

token identifier into a dense vector called an *embedding*.

Let the model dimension be d_{model}. An embedding matrix

$$E \in \mathbb{R}^{V \times d_{\text{model}}} \tag{1.4}$$

stores one vector for each vocabulary item. For token i, the corresponding representation is

$$e_i = E[i] \in \mathbb{R}^{d_{\text{model}}}. \tag{1.5}$$

These vectors are learned during training. Their dimensions do not usually correspond to simple human-readable properties. Meaning is distributed across many coordinates and, more importantly, evolves through successive Transformer layers.

Words or subwords that appear in related contexts often acquire geometrically related representations. This does not mean that the model contains a dictionary definition in vector form. Rather, it learns a high-dimensional organization that supports predictive computation.

1.3.1 Static and contextual representations

The initial embedding of a token is context-independent: the same vocabulary item begins with the same learned vector. The Transformer then converts it into a contextual representation. Thus the token `bank` develops different internal states in

> The bank approved the loan.

and

> They sat on the bank of the river.

The distinction emerges because attention combines the token with surrounding information.

1.4 Position and Sequence Order

A collection of token embeddings without position information is insufficient. The sentences

> The dog chased the child.

and

> The child chased the dog.

contain almost the same lexical material but express different relations.

Transformers therefore incorporate positional information. In a simplified additive form,

$$h_i^{(0)} = e_i + p_i, \tag{1.6}$$

where e_i is the token embedding and p_i encodes position i.

The original Transformer used sinusoidal positional encodings. Later systems adopted learned absolute embeddings, relative position methods, Rotary Positional Embeddings (RoPE), ALiBi, and other schemes. The engineering objective is not merely to label positions but to allow attention to represent relative distance and order over long contexts.

Technical note: long context is not perfect memory

A larger context window increases the maximum quantity of material that can be supplied to the model. It does not guarantee equal use of every token. Relevant information may be diluted, weakly attended, contradicted, or lost among distractors. Context capacity and context utilization are different properties.

1.5 The Transformer Architecture

The Transformer architecture, introduced by Vaswani et al. [32], replaced recurrence with attention-based sequence processing. Most generative

LLMs use a decoder-only Transformer: a stack of repeated blocks that process the current context while enforcing causal direction.

A typical block contains:

1. normalization;
2. masked multi-head self-attention;
3. a residual connection;
4. another normalization stage;
5. a position-wise feed-forward network;
6. another residual connection.

Architectural details vary. Some models use LayerNorm, others RMSNorm; some use dense feed-forward networks, others Mixture-of-Experts layers; some normalize before a sublayer, others after it. Nevertheless, the broad computational pattern remains recognizable.

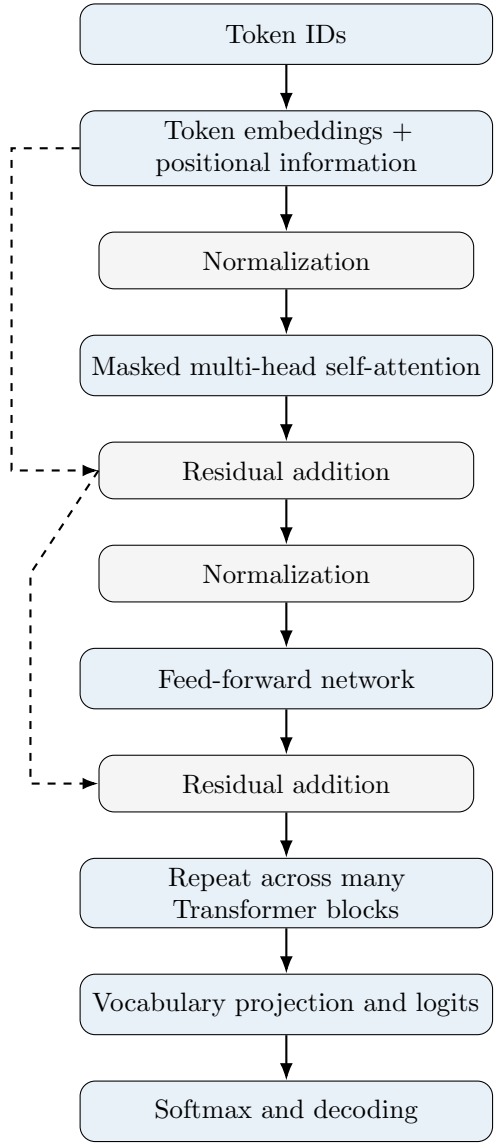

Figure 1.1: Simplified data flow in a decoder-only Transformer. Residual paths are shown as dashed arrows.

1.6 Self-Attention: Context as a Computation

Self-attention enables each token position to combine information from other positions in the same sequence. The mechanism begins by project-

ing each hidden state into three vectors:

$$Q = XW_Q, \tag{1.7}$$

$$K = XW_K, \tag{1.8}$$

$$V = XW_V, \tag{1.9}$$

where $X \in \mathbb{R}^{n \times d_{\text{model}}}$ is the matrix of hidden states, and W_Q, W_K, and W_V are learned projection matrices.

The attention operation is

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V. \tag{1.10}$$

The dot product $QK^\top$ measures compatibility between each query and each key. Division by $\sqrt{d_k}$ prevents the magnitude of dot products from growing excessively with dimensionality. The softmax converts scores into normalized weights, and those weights determine how value vectors are aggregated.

A useful intuition is:

- a *query* expresses what the current position is looking for;
- a *key* expresses what each candidate position offers for matching;
- a *value* carries the information that will be blended into the result.

This analogy is only approximate. Queries, keys, and values are learned numerical representations, not explicit semantic labels.

1.6.1 A small numerical example

Consider three tokens represented by simplified two-dimensional hidden states. Suppose the query for the current token is

$$q = \begin{bmatrix} 1 & 0 \end{bmatrix}, \tag{1.11}$$

and the keys are

$$k_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad k_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad k_3 = \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{1.12}$$

The unscaled compatibility scores are

$$qk_1^\top = 1, \qquad qk_2^\top = 0, \qquad qk_3^\top = 1. \tag{1.13}$$

With $d_k = 2$, scaled scores become

$$s = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \approx \begin{bmatrix} 0.707 & 0 & 0.707 \end{bmatrix}. \tag{1.14}$$

Applying softmax yields approximately

$$\alpha = \text{softmax}(s) \approx \begin{bmatrix} 0.401 & 0.198 & 0.401 \end{bmatrix}. \tag{1.15}$$

Let the values be

$$v_1 = \begin{bmatrix} 2 & 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0 & 3 \end{bmatrix}, \quad v_3 = \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{1.16}$$

The attention output is the weighted sum

$$o = 0.401v_1 + 0.198v_2 + 0.401v_3 \tag{1.17}$$

$$\approx \begin{bmatrix} 1.203 & 0.995 \end{bmatrix}. \tag{1.18}$$

The current position has therefore constructed a new representation by combining information from all three value vectors, with positions 1 and 3 receiving greater weight.

Worked example

In an actual LLM, vectors may contain thousands of dimensions, attention is computed for every relevant token pair, and multiple heads perform different projections in parallel. The numerical example is intentionally tiny, but the algebra is the same.

1.7 Causal Masking

An autoregressive model must not use future tokens when predicting the next token. During training, the full sequence is available computationally, so a causal mask blocks attention to positions to the right.

Let M be the mask

$$M_{ij} = \begin{cases} 0, & j \le i, \\ -\infty, & j > i. \end{cases} \tag{1.19}$$

Masked attention becomes

$$\text{MaskedAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right) V. \tag{1.20}$$

After softmax, entries associated with $-\infty$ receive zero probability. Position i can attend only to itself and earlier positions.

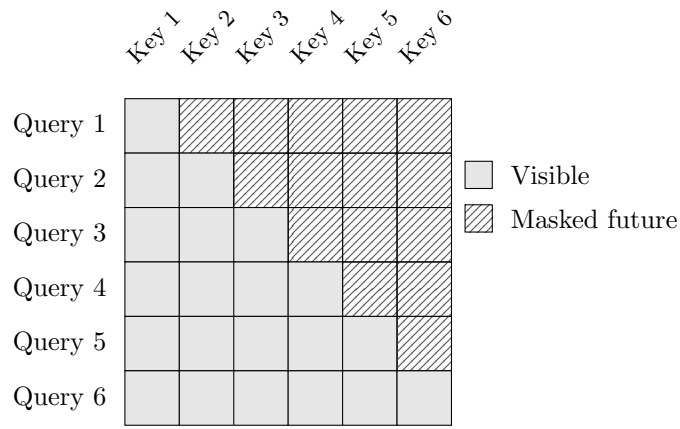

Figure 1.2: Triangular causal attention mask. Each query position can attend only to itself and earlier keys.

1.8 Multi-Head Attention

A single attention projection would constrain the model to one representation of relevance. Multi-head attention performs several attention

operations in parallel:

$$\text{head}_r = \text{Attention}(XW_Q^{(r)}, XW_K^{(r)}, XW_V^{(r)}), \tag{1.21}$$

for heads $r = 1, \ldots, h$. Their outputs are concatenated and projected:

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O. \tag{1.22}$$

Different heads can become sensitive to different relationships: local syntax, pronoun reference, delimiter matching, positional structure, topic continuity, or other distributed patterns. It would be misleading, however, to claim that every head has one stable human-interpretable function. Many computations are polysemantic and distributed.

1.9 Feed-Forward Networks and Feature Transformation

Attention mixes information across token positions. The feed-forward network transforms the representation at each position independently, using the same learned weights at every position.

A conventional form is

$$\text{FFN}(x) = W_2\,\sigma(W_1 x + b_1) + b_2, \tag{1.23}$$

where W_1 expands the vector into a larger intermediate dimension, σ is a nonlinear activation, and W_2 projects it back to the model dimension.

Modern LLMs often use gated variants such as SwiGLU. A simplified gated form is

$$\text{SwiGLU}(x) = W_2\left[\text{SiLU}(W_a x) \odot (W_b x)\right], \tag{1.24}$$

where $\odot$ denotes element-wise multiplication.

Feed-forward layers account for a substantial fraction of model parameters. They contribute to feature construction, transformation, and storage of distributed associations.

1.10 Residual Connections and Normalization

Deep neural networks are difficult to optimize if every layer must completely rewrite the representation. Residual connections allow a sublayer to learn an update:

$$y = x + F(x). \tag{1.25}$$

This creates a path along which information and gradients can flow more directly.

Normalization stabilizes activation scales. Layer Normalization is commonly written as

$$\text{LayerNorm}(x) = \gamma \odot \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta, \tag{1.26}$$

where μ and σ^2 are computed across features, and γ and β are learned parameters. RMSNorm removes mean centering and normalizes by root mean square.

In a pre-normalized block, a simplified sequence is

$$x' = x + \text{Attention}(\text{Norm}(x)), \tag{1.27}$$

$$y = x' + \text{FFN}(\text{Norm}(x')). \tag{1.28}$$

These mechanisms are engineering details with conceptual significance: the final behavior of an LLM is the cumulative effect of many small residual transformations rather than one explicit symbolic reasoning module.

1.11 From Hidden States to Next-Token Probabilities

After the final Transformer block, the hidden state at the last relevant position is projected into the vocabulary space:

$$z = hW_{\text{vocab}} + b, \tag{1.29}$$

where $z \in \mathbb{R}^V$ contains one score per vocabulary token. These scores are called *logits*. Softmax converts them to probabilities:

$$P(x_i \mid x_{<t}) = \frac{\exp(z_i)}{\sum_{j=1}^{V} \exp(z_j)}. \tag{1.30}$$

The model has now produced a distribution, not yet a final word. A decoding strategy chooses the next token. That token is appended to the sequence, and the process repeats.

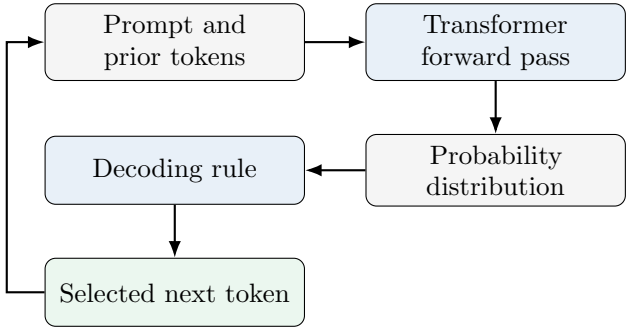

Figure 1.3: Autoregressive generation loop. The selected token is appended to the context and processed again.

1.12 Decoding: Determinism, Diversity, and Risk

The probability distribution can be converted into text in several ways.

1.12.1 Greedy decoding

Greedy decoding selects the highest-probability token:

$$x_t = \arg\max_x P(x \mid x_{<t}). \tag{1.31}$$

It is deterministic under fixed conditions but can produce repetitive or locally optimal text.

1.12.2 Temperature

Temperature rescales logits before softmax:

$$P_T(x_i) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}. \tag{1.32}$$

For $T < 1$, the distribution becomes sharper; for $T > 1$, it becomes flatter. Higher temperature increases diversity but can also increase instability. Lower temperature reduces randomness but does not guarantee truth. If the most probable continuation is false, greedy or low-temperature decoding may reproduce it consistently.

1.12.3 Top-k and nucleus sampling

Top-k sampling restricts selection to the k most probable tokens. Nucleus, or top-p, sampling selects the smallest set whose cumulative probability reaches a threshold p. These methods remove low-probability tails while retaining controlled variation.

Hallucination risk

Decoding parameters affect how the model explores its probability distribution. They influence variation, creativity, and consistency, but they do not add an independent factual verification mechanism. A false answer may be generated at any temperature, including zero-like deterministic settings.

1.13 Pretraining: Learning by Prediction

During pretraining, the model processes large corpora and repeatedly predicts held-out next tokens. The standard objective is cross-entropy loss:

$$\mathcal{L}(\theta) = -\sum_{t=1}^{n} \log P_\theta(x_t \mid x_{<t}), \tag{1.33}$$

where θ denotes all model parameters.

If the model assigns high probability to the observed token, the loss is low. If it assigns low probability, the loss is high. Backpropagation

computes gradients

$$\nabla_\theta \mathcal{L}, \tag{1.34}$$

and an optimizer updates the parameters. In simplified gradient-descent form,

$$\theta_{s+1} = \theta_s - \eta \nabla_\theta \mathcal{L}, \tag{1.35}$$

where η is the learning rate.

Real training uses mini-batches, adaptive optimizers, learning-rate schedules, parallel accelerators, mixed precision, gradient clipping, distributed communication, checkpointing, and many additional techniques. The conceptual point remains: the model is trained through prediction error, not by inserting a curated list of verified propositions into an explicit symbolic database.

1.13.1 Teacher forcing

During training, the model generally conditions on the true previous tokens rather than its own sampled outputs. This is often called teacher forcing. At inference time, however, each newly generated token becomes part of the next input. An early error can therefore alter the trajectory of subsequent generation.

Hallucination link: error propagation

Because generation is autoregressive, a weak or false early choice can become context for later tokens. The model may then elaborate the initial error into a coherent narrative, fabricated citation, or detailed explanation. Fluency can increase after the factual trajectory has already diverged.

1.14 Parameters, Capacity, and Distributed Knowledge

A parameter is a learned numerical value in the model's matrices and normalization layers. Modern LLMs may contain billions of parameters. Parameter count is not equivalent to the number of stored facts, and a fact is generally not located in one identifiable parameter.

Knowledge is distributed across many weights, layers, and activation patterns. A statement such as "Rome is the capital of Italy" may be supported by multiple textual associations, multilingual forms, geographical relations, and contextual pathways.

This distributed representation enables generalization. It also creates limitations:

- facts are not necessarily associated with explicit provenance;
- conflicting sources can be blended;
- rare knowledge can be weakly represented;
- updates are difficult to perform selectively;
- memorization and generalization can coexist;
- the same information may be elicited differently by different prompts.

Technical note: parameters, tokens, FLOPs, and context

Parameters

Learned numerical values that determine the model's transformations. They are a rough indicator of capacity, not a direct measure of intelligence or factual accuracy.

Training tokens

Token instances processed during training. More data can improve coverage, but quality, duplication, contamination, and domain balance matter.

FLOPs

Floating-point operations used as an approximation of com-

putational cost. Training compute depends on model size, token count, architecture, hardware efficiency, and optimization strategy.

Context window
Maximum token span available during one inference episode. It is temporary working context, not equivalent to permanent learning.

KV cache
Stored key and value tensors from previous decoding steps, used to avoid recomputing attention over the entire prefix for every new token.

1.15 Scaling Laws and Capability Growth

Empirical scaling laws show that language-model loss tends to improve predictably with increases in model size, data, and compute under suitable regimes [16, 8]. These relationships guide decisions about how to allocate a fixed training budget.

Scaling can improve language quality, few-shot performance, code generation, multilingual transfer, and complex task completion. Some capabilities appear to emerge sharply at particular scales, although apparent discontinuities may partly reflect thresholded evaluation metrics.

Scale should not be confused with epistemic reliability. A larger model may possess broader knowledge and stronger reasoning patterns, but it can also produce more persuasive falsehoods. The same capacity that improves explanation and synthesis can improve the coherence of hallucinated content.

1.16 Instruction Tuning and Preference Optimization

A pretrained model is a continuation engine. It may continue a question with another question or imitate the style of its training data rather

than behave as a helpful assistant. Post-training aligns the model with conversational tasks.

1.16.1 Supervised fine-tuning

Supervised fine-tuning trains the model on instruction-response pairs:

$$(x_{\text{instruction}}, y_{\text{preferred}}). \tag{1.36}$$

The objective remains token prediction, but the examples teach the desired interaction format, tone, and task behavior.

1.16.2 Preference learning

Human or synthetic evaluators can rank candidate responses. A reward model may estimate

$$r_\phi(x, y), \tag{1.37}$$

where x is a prompt and y a response. Reinforcement Learning from Human Feedback (RLHF) uses such preferences to optimize behavior [26]. Direct Preference Optimization (DPO) and related methods optimize preference relationships more directly [27].

Post-training can improve helpfulness, safety, instruction following, and conversational quality. Yet it can create a subtle tension: evaluators often prefer complete, confident, well-structured answers. Unless uncertainty and abstention are explicitly rewarded, the model may learn that sounding useful is safer than admitting insufficient knowledge.

Hallucination link: helpfulness pressure

A system rewarded for producing an answer may infer that a plausible completion is preferable to silence. Hallucination risk therefore depends not only on pretraining data but also on the behavioral incentives introduced during post-training.

1.17 In-Context Learning

LLMs can adapt to examples and instructions placed directly in the prompt without changing their parameters. This is called in-context learning.

A few-shot prompt may contain

```
English: red
Italian: rosso

English: blue
Italian: blu

English: green
Italian:
```

The model infers the local task and continues with a likely translation. Zero-shot prompting supplies only an instruction, while one-shot and few-shot prompting include one or more demonstrations.

In-context learning is not identical to permanent learning. The parameter vector remains unchanged. The model constructs a temporary computation from the supplied sequence.

This flexibility explains why prompt wording can have large effects. The prompt influences task interpretation, assumptions, format, reasoning style, and confidence. It can also implant false premises. If a user asks, "Why did the 2028 Treaty of Lyon abolish quantum patents?", a model may accept the presupposition and generate an explanation rather than challenge the existence of the treaty.

1.18 Context Windows, Attention Limits, and Memory

The active context can include system instructions, user messages, previous answers, retrieved documents, tool outputs, and structured data. It functions as working material for the current computation.

A long context does not imply human-like memory. Several failure modes remain possible:

- earlier information is truncated;
- relevant evidence is present but underweighted;
- two passages conflict and the model blends them;
- the model follows a recent instruction while neglecting an earlier constraint;
- a summary removes a detail needed later;
- distractor text shifts attention away from the correct source.

The model's apparent memory is therefore the product of architecture, context management, retrieval, summarization, and application design—not one unified faculty.

1.19 External Knowledge, Retrieval, and Tools

A standalone LLM relies primarily on parametric knowledge and current context. A broader AI system can connect the model to external components:

Retrieval systems
: Search documents, databases, or vector indexes and insert relevant passages into the prompt.

Tools
: Execute calculations, code, database queries, web searches, or enterprise actions.

Structured memory
: Store facts, preferences, conversation summaries, or workflow state outside the model weights.

Verification modules
: Check citations, constraints, schemas, numerical outputs, or business rules.

Retrieval-Augmented Generation (RAG) can be summarized as

$$\text{question} \rightarrow \text{retrieval} \rightarrow \text{augmented context} \rightarrow \text{generation}. \tag{1.38}$$

The method can improve freshness, traceability, and domain specificity [18]. Nevertheless, RAG is not a guarantee of truth. The retriever may return the wrong passage; the source may be outdated; the model may misread it; or the generated answer may contain unsupported additions.

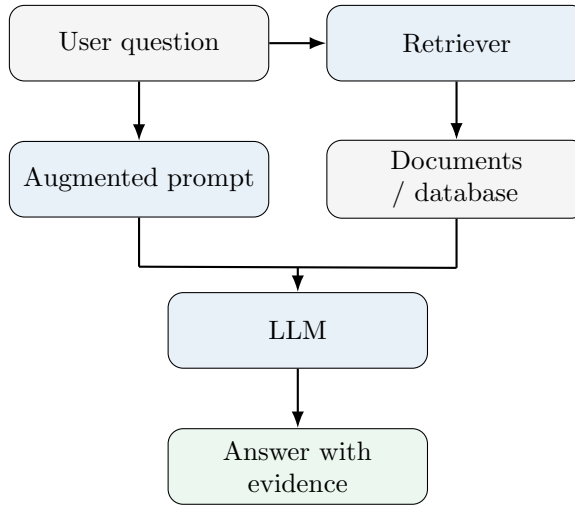

Figure 1.4: Simplified Retrieval-Augmented Generation pipeline. Each component can introduce its own failure modes.

1.20 Mixture-of-Experts Models

Some LLMs replace every dense feed-forward layer with a Mixture-of-Experts (MoE) mechanism. A router selects a small subset of expert networks for each token. If there are m experts but only k are activated per token, the model can have a large total parameter count while using a smaller active computation.

In simplified form,

$$y = \sum_{r \in \mathcal{T}(x)} g_r(x) E_r(x), \tag{1.39}$$

where $\mathcal{T}(x)$ is the selected set of experts, $g_r(x)$ is the routing weight, and E_r is expert r.

MoE architectures improve capacity-compute trade-offs, but they introduce routing, load-balancing, and specialization challenges. They do not fundamentally change the probabilistic nature of next-token generation.

1.21 Inference Efficiency and the KV Cache

During autoregressive generation, naively recomputing attention for the entire prefix at every step would be expensive. Transformers therefore cache the key and value tensors produced for previous tokens. This *KV cache* allows the model to compute only the new query, key, and value contributions at each decoding step.

The cache reduces repeated computation but consumes memory proportional to sequence length, number of layers, attention dimensions, and batch size. Techniques such as multi-query attention, grouped-query attention, quantization, speculative decoding, and paged attention are used to improve inference efficiency.

These mechanisms affect latency and cost rather than the conceptual truth criterion. A faster model remains a probabilistic generator.

1.22 Does an LLM Understand?

The question of understanding has at least three meanings:

Behavioral understanding

The system responds in ways associated with competent understanding.

Representational understanding

Internal states encode semantic, syntactic, relational, or task-

relevant structure.

Phenomenological understanding

The system has subjective experience or conscious awareness of meaning.

LLMs provide substantial evidence for the first and aspects of the second in an operational sense. Their representations support abstraction, analogy, translation, classification, and flexible task performance. These observations do not establish the third.

The distinction matters because users often interpret conversational fluency as evidence of a mental state. When the model outputs a false statement, they may say that it "knew" the truth and "lied." Such language imports intention into a mechanism whose immediate operation is probabilistic token generation.

This does not make the output harmless. A system can mislead without possessing human-like intention. The absence of intent changes the philosophical classification, not the practical consequences.

1.23 Why Fluency Is Not Evidence of Truth

Suppose a model generates the citation

> J. Smith and R. Taylor, "Neural Veracity in Generative Systems," *Journal of Computational Cognition*, vol. 18, no. 2, 2019.

The entry is structurally plausible. It contains familiar author names, an academic title, a journal-like venue, volume and issue numbers, and a reasonable year. Yet the work may not exist.

The model can construct such a citation because it has learned the statistical form of references and associations among domain terms. It is effectively optimizing

$$\arg\max_{y} P(y \mid x), \tag{1.40}$$

not

$$\arg\max_{y} \text{Truth}(y). \tag{1.41}$$

The second expression is not an ordinary differentiable objective available to the standalone model. Truth requires a relation among a proposition, evidence, context, time, and the external world. Next-token likelihood is computed within the model's learned distribution.

Key idea

Fluency is evidence that the model has learned the form of competent language. It is not, by itself, evidence that the generated proposition has been checked against reality.

1.24 The Technical Path to Hallucination

The architecture described in this chapter creates several pathways to hallucination:

1. **Incomplete parametric knowledge.** The model has only weak or conflicting representations of the relevant fact.
2. **Prompt pressure.** The question presupposes an answer or requests unsupported precision.
3. **Pattern completion.** The model produces a structurally plausible continuation even when evidence is absent.
4. **Autoregressive commitment.** An early error becomes context for later elaboration.
5. **Context failure.** Relevant information is missing, truncated, diluted, or misinterpreted.
6. **Retrieval failure.** External evidence is wrong, stale, or irrelevant.
7. **Alignment pressure.** Post-training rewards helpfulness, completeness, or confidence more strongly than abstention.
8. **Decoding variation.** Sampling selects a low-probability but

locally coherent branch.

9. **No independent verifier.** Generation completes without a separate factual check.

These factors can produce an output that is grammatically correct, semantically coherent, stylistically authoritative, and factually false.

1.25 A Layered Model of an LLM System

It is useful to distinguish the model itself from the complete product in which it operates.

Table 1.1: Layers of a production LLM system and representative failure modes.

Layer	**Primary role**	**Possible failure**
Tokenizer	Converts text to token IDs	Poor handling of rare names, numbers, scripts, or formatting
Model weights	Encode learned predictive structure	Outdated, incomplete, biased, or conflicting knowledge
Context manager	Selects conversation and document content	Truncation, omission, or instruction conflict
Retriever	Finds external evidence	Irrelevant, stale, or low-quality sources
Prompt orchestration	Frames the task and constraints	Leading assumptions, ambiguous goals, or weak grounding
Decoder	Selects next tokens	Excessive randomness, repetition, or unstable trajectories
Tool layer	Executes searches, code, or actions	Tool misuse, incorrect parameters, or unverified results
Post-processing	Formats and validates outputs	Broken citations, schema errors, or omitted warnings
Human workflow	Reviews or acts on the answer	Automation bias, insufficient expertise, or lack of escalation

Calling every failure an "LLM hallucination" can therefore conceal the true cause. Some failures originate in retrieval, orchestration, stale enterprise data, user assumptions, or downstream software. Precise diagnosis requires identifying the layer at which evidence was lost or distorted.

1.26 The Pinocchio Effect

The Pinocchio metaphor is attractive because it provides an immediate image of falsehood. Yet the metaphor must be bounded carefully. Pinocchio's nose grows when he lies; lying ordinarily implies awareness of truth and an intention to deceive. An LLM can generate false content without either property.

The *Pinocchio Effect* may therefore be defined as follows:

Working definition

The **Pinocchio Effect** is the human perception of intentional or knowledgeable deception produced when an AI system generates false or unsupported content in a fluent, confident, and socially persuasive form.

The effect arises from a mismatch between two systems:

- the machine produces language through probabilistic computation;
- the human interprets coherent language through social and intentional categories.

Language is normally evidence of a speaker. A speaker is normally assumed to possess beliefs, intentions, memory, and responsibility. Conversational AI activates these assumptions even when its internal mechanism differs radically from human speech production.

The "growing nose" is therefore not located only in the model. It appears in the interaction among architecture, interface, user expectation, institutional trust, and the absence of visible uncertainty.

1.27 Chapter Summary

A modern LLM can be understood as a layered probabilistic system:

1. Text is converted into tokens.

2. Tokens are mapped to vectors and combined with position information.
3. Transformer blocks repeatedly apply masked self-attention and feed-forward transformations.
4. The final hidden state is projected into vocabulary logits.
5. Softmax produces a probability distribution over the next token.
6. A decoding strategy selects one token.
7. The token is appended, and the cycle repeats.
8. Pretraining adjusts parameters to minimize next-token prediction loss.
9. Post-training shapes the model into an instruction-following assistant.
10. Retrieval, tools, memory, and verification may extend the system but also introduce new failure modes.

The model's strength and vulnerability emerge from the same source. To predict language well, it learns rich representations of the patterns through which humans express knowledge. But the generation process does not inherently distinguish a verified fact from a highly plausible fabrication.

The next chapter must therefore ask a more precise question than "Why does the AI lie?" It must ask: *what exactly is an AI hallucination, how should it be classified, and when does a generative error become an epistemic and operational risk?*

Technical Checklist

After reading this chapter, the reader should be able to explain:

- why an autoregressive LLM factorizes sequence probability token by token;
- how tokenization differs from word segmentation;
- what embeddings and contextual hidden states represent;
- how queries, keys, and values produce attention weights;
- why causal masking is required for next-token prediction;
- how multi-head attention and feed-forward layers differ;
- how logits, softmax, temperature, top-k, and top-p affect generation;
- how cross-entropy and backpropagation train the model;
- why parameter count is not a count of stored facts;
- how instruction tuning and preference optimization alter behavior;
- why long context, RAG, and tools reduce some risks but do not guarantee truth;
- why fluency can create the perception of knowledge and intention.

2

What Is an AI Hallucination?

A hallucination is not merely a wrong sentence. It is a failure of grounding: language continues after evidence has ended, while the surface of the answer conceals the break.

2.1 The Difficulty of Naming the Failure

The expression *AI hallucination* is now used for a wide family of failures: invented facts, false citations, contradictions with a supplied document, fabricated intermediate reasoning, imaginary software functions, unsupported visual descriptions, and confident answers to questions that cannot be answered from the available evidence. The term is useful because it identifies a recognizable phenomenon. It is also dangerous because it can become so broad that it explains nothing.

A precise study must begin by separating three questions:

1. Is the output false with respect to the external world?
2. Is the output unsupported by, or contradictory to, the information supplied to the model?

3. Does the output violate the task, conversational state, or constraints under which it was generated?

These questions overlap, but they are not equivalent. A sentence can be factually true and still be unfaithful to a source. A sentence can faithfully reproduce a source that is itself false. A response can be factually correct but fail the user's request. A chain of reasoning can contain an invalid step even though its final answer happens to be correct.

Research literature consequently offers multiple definitions and taxonomies rather than one universally accepted boundary. Surveys of natural-language generation and large language models generally converge on the idea that hallucination involves generated content that is unsupported, inconsistent, unverifiable, or false relative to an appropriate reference[12, 9]. The difficulty lies in deciding what the appropriate reference is.

Key idea

A useful definition of hallucination must specify the **reference frame**: the input document, retrieved evidence, external reality, prior dialogue, tool output, task specification, or some combination of these.

2.2 A Working Definition

For this book, an AI hallucination is defined as follows:

Working definition

An **AI hallucination** is generated content that is presented as an answer, description, explanation, or inference but lacks adequate support from the relevant evidence, contradicts that evidence, or asserts a proposition that is false or unjustifiably specific under the conditions of the task.

This definition contains five elements.

Generated content

The phenomenon concerns output produced by a generative system, not merely an error stored in a database or entered by a human.

Assertoric presentation

The content is offered as though it describes, explains, or follows from something. Fiction requested as fiction is not a hallucination merely because it is unreal.

Relevant evidence

Support must be judged against the evidence appropriate to the task, not against an undefined totality of knowledge.

Epistemic defect

The content is unsupported, contradictory, false, or more precise than the evidence allows.

Task conditions

Time, domain, source constraints, user intent, and the system's access to tools all affect the judgment.

The phrase *adequate support* is deliberate. Evidence is rarely binary in practical systems. A source may support part of a claim, support it only indirectly, or support it under assumptions that the model fails to disclose. Hallucination analysis therefore requires both logical judgment and contextual interpretation.

2.3 Hallucination, Confabulation, and Fabrication

The word *hallucination* originates in human perceptual and clinical contexts. Applied to AI, it is metaphorical. The model does not necessarily experience a perception that is absent from reality. It produces an output through computation.

Some institutions prefer the term *confabulation*. The United States National Institute of Standards and Technology describes confabulation as generative AI producing confidently presented erroneous or false

content, including outputs that diverge from inputs or contradict earlier statements in the same context[25]. This terminology emphasizes fluent invention rather than sensory experience.

Fabrication is also common, especially when the output invents a source, person, event, quotation, or numerical result. However, fabrication can imply purposeful construction, and in ordinary speech may suggest intent. A model can fabricate without intending to deceive.

No term is perfect. Each carries a metaphor:

Table 2.1: Common terms and their conceptual limitations.

Term	**What it emphasizes**	**Possible misunderstanding**
Hallucination	Output detached from reality or evidence	Suggests human-like perception or mental illness
Confabulation	Fluent filling of gaps without reliable memory or evidence	Still borrowed from clinical psychology and neurology
Fabrication	Construction of details, entities, sources, or events	May imply deliberate invention
Factual error	Incorrect proposition	Too narrow for true-but-unsupported or context-contradicting output
Ungrounded generation	Lack of support in an input or evidence set	May omit open-world factual falsehoods when no source is supplied
Falsehood	A proposition that is not true	Does not capture unverifiable, misleading, or task-inconsistent output

This book retains *hallucination* because it is established in technical and public discourse, while using *confabulation*, *fabrication*, *factual error*,

and *ungrounded generation* when those terms are more precise.

Boundary case

Terminology should not become a substitute for diagnosis. The practical question is not only "Was this a hallucination?" but also "What reference was violated, at what granularity, through which system component, and with what consequence?"

2.4 The Three Anchors of Grounding

A generated response can be evaluated against three principal anchors: the supplied evidence, the external world, and the task state.

Let

- x denote the prompt and supplied context,
- r denote retrieved or tool-produced evidence,
- W_t denote the relevant state of the world at time t,
- τ denote the task specification,
- y denote the generated response.

The output may fail in at least three ways:

$$H_{\text{source}}(y): \quad y \not\models (x, r), \tag{2.1}$$

$$H_{\text{world}}(y): \quad y \not\models W_t, \tag{2.2}$$

$$H_{\text{task}}(y): \quad y \not\models \tau. \tag{2.3}$$

The symbol $\models$ is used informally here to represent compatibility or support, not strict logical entailment in every case.

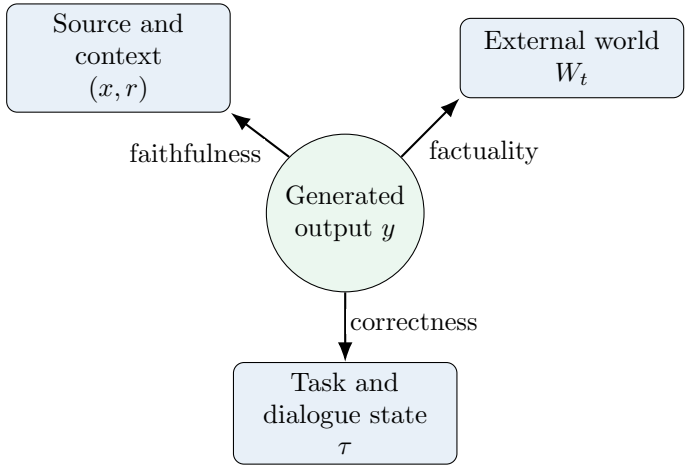

Figure 2.1: The three principal anchors against which generated content can be grounded.

2.4.1 Source grounding

In summarization, document question answering, retrieval-augmented generation, and data-to-text tasks, the supplied material is the primary reference. An output is unfaithful when it contradicts or exceeds what the source supports.

2.4.2 World grounding

In open-domain question answering, biography generation, scientific explanation, or current-affairs discussion, the relevant reference is external reality at an appropriate time. The output may be fluent and internally coherent while being factually false.

2.4.3 Task grounding

A model may correctly state facts but violate the requested constraints. It may answer the wrong question, attribute an intention the user did not express, use a prohibited data source, or silently change units. Some researchers treat these as instruction-following failures rather than hallucinations. In production systems, however, the boundary matters

less than the operational fact that the system asserts something not warranted by the task.

2.5 Faithfulness Is Not the Same as Factuality

The distinction between *faithfulness* and *factuality* is central.

Faithfulness

The output is supported by, or correctly derived from, the designated input or evidence.

Factuality

The output corresponds to the relevant external reality.

Research in abstractive summarization showed that generated text can introduce information absent from the source, and that some such additions can nevertheless be factually correct[23, 3]. This produces four logical possibilities.

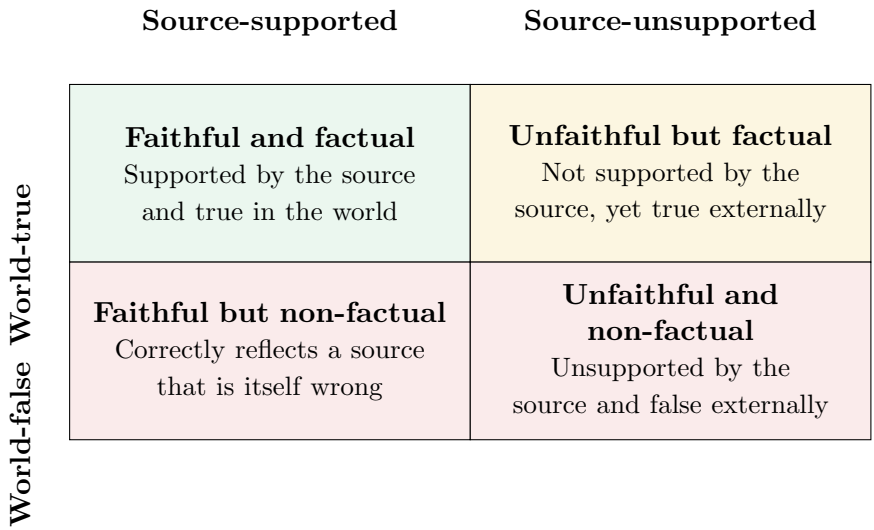

Figure 2.2: Faithfulness and factuality form distinct evaluation dimensions.

Example: true but unfaithful

Source: "The company opened a new plant in Turin in 2024."
Generated summary: "The company, founded in 1958, opened a new plant in Turin in 2024."
Assume the founding year is externally correct. The summary is factual in the open world but unfaithful to the designated source because the source does not supply the year. In a source-bounded task, the added detail should still be flagged.

Example: faithful but false

Source: "The Moon is a planet."
Generated summary: "The source states that the Moon is a planet."
The output can be faithful as a report of what the source says while the embedded proposition is false. A well-designed system should preserve this distinction through attribution rather than silently endorsing the claim.

2.6 Intrinsic and Extrinsic Hallucinations

A widely used taxonomy distinguishes *intrinsic* from *extrinsic* hallucinations[23, 12].

2.6.1 Intrinsic hallucination

An intrinsic hallucination contradicts or distorts information present in the source. The model uses source material but changes a relation, entity, polarity, quantity, or event.

Intrinsic contradiction

Source: "The regulator approved the merger after a six-month review."

Output: "The regulator rejected the merger after a six-month review."
The entities and event are inherited from the source, but the decision is reversed.

Intrinsic hallucinations often arise through:

- entity swapping;
- predicate reversal;
- negation loss;
- incorrect coreference;
- attachment of a correct number to the wrong event;
- causal inversion;
- temporal reordering.

2.6.2 Extrinsic hallucination

An extrinsic hallucination introduces content that cannot be verified from the designated source. The added content may be false, true, or unverifiable from the available evidence.

Extrinsic addition

Source: "The clinical trial enrolled 420 participants."
Output: "The clinical trial enrolled 420 participants across twelve hospitals."
The number of hospitals is absent from the source. Even if twelve happens to be correct, the statement is ungrounded relative to the supplied evidence.

The intrinsic/extrinsic distinction is valuable for source-conditioned tasks but insufficient for open-domain generation. When no source is

supplied, hallucination must be judged against external knowledge, task constraints, or consistency.

2.7 A Broader Taxonomy for Large Language Models

Large language models operate across more diverse tasks than traditional summarization systems. A broader taxonomy is therefore required.

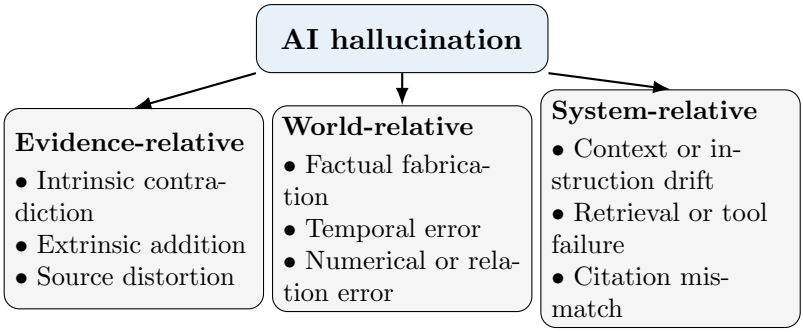

Figure 2.3: A high-level taxonomy. Categories can overlap in a single response.

2.7.1 Entity hallucination

The output invents or substitutes a person, organization, product, location, chemical, legal instrument, software library, or other named entity.

Examples include a nonexistent professor, a fictitious court, an imaginary API method, or a company attributed to the wrong country.

2.7.2 Attribute hallucination

The entity exists, but the model assigns an unsupported property: date of birth, job title, diagnosis, material composition, ownership, capacity, or performance value.

2.7.3 Relation hallucination

The entities and attributes may be real, but the relationship is false. The model may state that one company acquired another, that a paper cites a particular theory, or that a medicine causes an outcome without adequate support.

2.7.4 Event hallucination

The output invents an event or changes its participants, location, time, sequence, or outcome. This category is common in news summaries, biographies, incident reports, and meeting recaps.

2.7.5 Numerical hallucination

The model produces a wrong quantity, percentage, unit, date, price, measurement, probability, or calculation. Numerical errors are especially persuasive because precision creates an impression of evidence.

Not every arithmetic mistake is best described as hallucination. A transparent calculation error may be a reasoning failure. It becomes hallucinatory when the model supplies an unsupported value, presents invented intermediate data, or confidently attributes a number to a source that does not contain it.

2.7.6 Temporal hallucination

A proposition can change truth value over time. Office-holders, laws, prices, software versions, product specifications, and organizational structures are time-sensitive. A statement that was correct during training may be false at deployment.

Let W_t denote the world at time t. Factuality should be evaluated as

$$\mathrm{True}(a, W_t), \tag{2.4}$$

not as an atemporal property. A response can therefore be historically accurate but currently wrong, or current but inappropriate for a question

explicitly about the past.

2.7.7 Citation and provenance hallucination

The model invents a paper, author, journal, URL, legal case, quotation, page number, or document identifier. A subtler form occurs when the source exists but does not support the claim attributed to it.

Citation hallucination should be decomposed into at least four checks:

a. Does the cited source exist?

b. Are the bibliographic details correct?

c. Does the source contain the attributed material?

d. Does the source actually support the conclusion drawn from it?

Hallucination risk

A real citation can create more false confidence than an obviously fictitious one. Verification must evaluate entailment and relevance, not merely source existence.

2.7.8 Quotation hallucination

The model may produce words that a person never said, paraphrase a source while placing quotation marks around the paraphrase, or combine fragments into a synthetic quotation. This is both a factual and provenance failure.

2.7.9 Logical or reasoning hallucination

The output contains an invalid inference, assumes a missing premise, changes a condition, or generates a persuasive explanation for an answer that is unsupported.

The term is contested here. In a narrow taxonomy, a reasoning error is distinct from hallucination. In a broad taxonomy, fabricated premises and unsupported inferential steps are hallucinations because they assert relations that have no evidential basis.

Boundary case: correct answer, invalid explanation

A model may produce the correct final answer through an invalid chain of reasoning. The result is task-correct but explanation-unfaithful. Conversely, a valid-looking explanation may end in a wrong answer because of one arithmetic operation. Evaluation must score the answer and the rationale separately.

2.7.10 Conversational hallucination

In multi-turn dialogue, a model may claim that the user previously stated something that was never said, confuse people or projects, contradict an earlier answer, or silently change an agreed constraint. This is a failure relative to dialogue state.

2.7.11 Intent hallucination

The system infers a goal, preference, emotion, or instruction that the user did not express. For example, it may answer a request for neutral analysis as though the user had requested advocacy. This can be understood as an instruction-following or pragmatic error, but it is operationally similar to hallucination because the system invents part of the task.

2.7.12 Retrieval and tool hallucination

In a tool-augmented system, the model may:

- claim to have searched when no search occurred;
- misread a tool response;
- cite a retrieved document that does not support the answer;
- merge values from different records;
- invent tool results after a failed call;
- state that an action was completed when it was not.

These are system-level failures. The model may be only one contributor. The retriever, parser, orchestration logic, stale database, or interface may be responsible for the loss of grounding.

2.7.13 Multimodal hallucination

Vision-language and audio-language systems can describe objects, text, speakers, sounds, or events not present in the input. They can also miss visible evidence, attach a real object to the wrong location, or infer an identity or activity beyond what the image supports.

Multimodal hallucination preserves the same structure as textual hallucination: an assertion exceeds or contradicts the available evidence. The reference, however, is a combination of modalities rather than text alone.

2.8 Hallucination Is Not the Same as Every Model Failure

A rigorous taxonomy requires exclusions.

2.8.1 Creative invention

When the user requests a fictional story, imaginary city, speculative scenario, or metaphor, invented content is the intended product. It becomes hallucinatory only when the model misrepresents fiction as fact, violates the fictional world's established constraints, or invents supposedly real evidence.

2.8.2 Omission

A summary can omit an essential fact without adding any false content. This is a completeness or coverage failure, not ordinarily a hallucination. Nevertheless, omission can become misleading when the remaining statements imply a false overall conclusion.

2.8.3 Irrelevance

An answer may be true but irrelevant. This is primarily a relevance or instruction-following defect. If the irrelevance results from the model inventing a different user intention, the failure may overlap with intent hallucination.

2.8.4 Bias

Bias and hallucination can interact but are conceptually different. A stereotype may be factually false and therefore hallucinatory in a specific instance, while systematic representational bias can exist even in factually accurate outputs.

2.8.5 Refusal and over-caution

A model that refuses an answer it could safely and correctly provide has failed utility or calibration, not hallucinated. Excessive abstention and excessive assertion are opposite errors.

2.8.6 Malicious deception

A system can be prompted or optimized to deceive. Deception concerns behavior that strategically causes false belief, often with an objective or policy-level notion of intent. Ordinary hallucination does not require such intent. Conflating the two obscures both.

Table 2.3: Hallucination and adjacent failure classes.

Failure	**Core defect**	**Relationship to hallucination**
Hallucination	Unsupported, contradictory, false, or unjustifiably specific assertion	Central category
Omission	Relevant information absent	Usually distinct; can create misleading implication
Irrelevance	Content does not address the task	Distinct unless based on invented intent or context
Reasoning error	Invalid derivation or computation	Overlaps when premises or explanations are fabricated
Bias	Systematic skew or unfair representation	Can coexist with factual accuracy or hallucination
Deception	Strategic production of false belief	Requires a stronger account of objective or intent
Uncertainty failure	Confidence does not match correctness	Amplifies hallucination but can occur with correct answers

2.9 From Sentences to Atomic Claims

A long response is rarely entirely true or entirely false. It often contains a mixture of supported and unsupported propositions. Binary passage-level labels conceal this structure.

Let an output y be decomposed into a set of atomic claims:

$$A(y) = \{a_1, a_2, \ldots, a_m\}. \tag{2.5}$$

An atomic claim should express one independently verifiable proposition. Consider:

> Ada Lovelace was born in London in 1815 and published the first complete computer program in 1843.

This sentence contains at least three claims:

1. Ada Lovelace was born in London.
2. Ada Lovelace was born in 1815.
3. Ada Lovelace published the first complete computer program in 1843.

Each claim can receive a support label:

$$S(a_i \mid E) \in \{\text{supported}, \text{contradicted}, \text{insufficient evidence}\}, \tag{2.6}$$

where E is the designated evidence set.

The distinction between *contradicted* and *insufficient evidence* matters. A claim can lack support without available evidence proving it false. Treating both as identical may inflate error estimates or encourage unjustified correction.

2.9.1 Claim precision

A simple factual precision measure is

$$\text{Precision}_{\text{fact}}(y) = \frac{\sum_{i=1}^{m} \mathbb{I}[a_i \text{ is supported}]}{m}. \tag{2.7}$$

FActScore formalizes a related approach by decomposing long-form generations into atomic facts and calculating the proportion supported by a reliable knowledge source[24].

2.9.2 Claim density

Longer answers can contain more errors simply because they make more claims. Hallucination evaluation should therefore consider both proportion and volume:

$$N_{\text{claims}} = m, \tag{2.8}$$

$$N_{\text{unsupported}} = \sum_{i=1}^{m} \mathbb{I}[a_i \text{ unsupported}], \tag{2.9}$$

$$R_{\text{unsupported}} = \frac{N_{\text{unsupported}}}{m}. \tag{2.10}$$

A response with 95 percent factual precision but 100 claims contains five unsupported claims. In a high-stakes domain, the absolute count may matter more than the percentage.

Granularity problem

The measured hallucination rate depends on how claims are segmented. A coarse claim may be marked false because one detail is wrong, while a fine decomposition may reveal that most components are supported. Evaluation protocols must define atomicity consistently.

2.10 Degrees of Support

Support is not always binary. A claim may be explicit, entailed, inferable, plausible, or merely compatible with evidence.

A practical ordinal scale is:

S1: Explicitly supported – directly stated in the evidence.

S2: Strongly entailed – follows through a short, uncontroversial inference.

S3: Conditionally supported – follows only under stated assumptions.

S4: Plausible but unsupported – compatible with the evidence but not established.

S5: Contradicted – incompatible with the evidence.

Hallucination begins clearly at S4 and S5 in a source-bounded task. S2 and S3 require task-specific policy. A legal summary, clinical record, and creative explanation tolerate different degrees of inference.

The same applies to truth. Some claims are well-established, some disputed, some time-dependent, and some inherently interpretive. An evaluator must not convert legitimate uncertainty into a falsely precise label.

2.11 Confidence, Calibration, and Epistemic Posture

Hallucination is often described as confidently stated falsehood. Confidence increases risk, but it is not part of every technical definition. A false answer written hesitantly remains false; a true answer written confidently may be appropriate.

The important concept is *calibration*. Suppose the model assigns confidence p_i to answer i, and $z_i = 1$ when the answer is correct. A calibrated system satisfies approximately

$$P(z_i = 1 \mid p_i = p) \approx p. \tag{2.11}$$

Among answers labelled 80 percent confident, roughly 80 percent should be correct. Research has found that model self-evaluation can carry useful information under some conditions, while calibration remains sensitive to task format, distribution shift, and prompting[13, 14].

Verbal confidence is not identical to probabilistic confidence. Phrases such as "certainly," "the evidence proves," or "the correct answer is" are stylistic tokens. They may not faithfully communicate internal uncertainty.

Hallucination risk

The most dangerous combination is not merely error, but **error plus low detectability**: fluent wording, precise detail, familiar formatting, and absent uncertainty markers.

2.12 Hallucination as a Generation Process

A hallucination is often evaluated as a property of the completed output, but it also develops as a trajectory.

At step t, the model samples or selects token x_t conditioned on the existing prefix. Once an unsupported detail enters the sequence, it becomes part of the context for later tokens:

$$P(x_{t+1} \mid x_{\leq t}). \tag{2.12}$$

The model may then elaborate the initial error with dates, causal explanations, quotations, and consequences. This is *autoregressive commitment*: later text becomes locally coherent with an earlier unsupported premise.

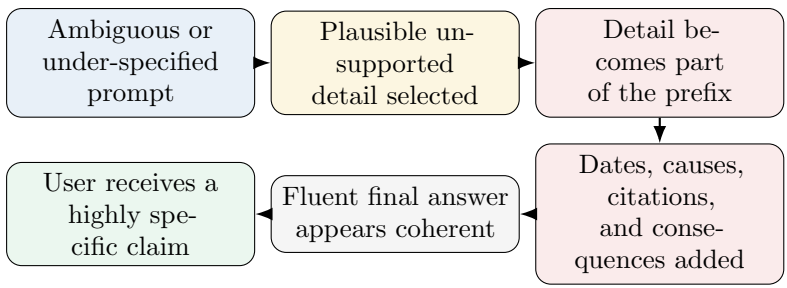

Figure 2.4: A simplified hallucination trajectory during autoregressive generation.

This process explains why an answer can become more detailed after the first error rather than self-correcting. Detail is not independent evidence. It may be the continuation of a mistaken branch.

2.13 Causes Are Not Definitions

Hallucination is defined by the relation between output and evidence or reality. Its cause is a separate question. The same observable failure can arise from different mechanisms:

- missing or conflicting training data;
- stale parametric knowledge;
- ambiguous prompts;
- misleading presuppositions in the question;
- retrieval of irrelevant evidence;
- context truncation;
- weak attention to a long source;
- decoding randomness;
- instruction tuning that rewards completion over abstention;
- tool failure or orchestration error;
- user-requested precision beyond available evidence.

Diagnosing cause from output alone is difficult. A fabricated citation could arise because the work was absent from training, because similar citations were blended, because the user supplied an incorrect title, or because a retrieval call failed. The surface form does not reveal the mechanism with certainty.

2.14 Measuring Hallucination

No single metric captures all hallucination types. Evaluation methods differ according to reference availability, task, domain, and granularity.

2.14.1 Human evaluation

Human annotators can inspect claims against source documents or authoritative references. This remains the strongest general method when experts, clear rubrics, and adequate time are available. Its weaknesses include cost, disagreement, fatigue, domain dependence, and inconsistent interpretation of support.

A strong annotation protocol should define:

- the reference evidence;
- the unit of annotation;
- labels for support, contradiction, and uncertainty;
- treatment of common knowledge;
- temporal cutoff;
- acceptable inference depth;
- rules for disputed claims;
- escalation to domain experts.

2.14.2 Entailment-based evaluation

Natural language inference models can estimate whether evidence entails, contradicts, or is neutral toward a claim. These systems are useful but inherit their own model errors. Long documents, numerical reasoning, negation, and domain-specific language remain difficult.

2.14.3 Question-answering-based evaluation

An evaluator generates questions from the output and checks whether the source yields compatible answers. This can reveal inconsistencies but depends on question quality and answer comparison.

2.14.4 Retrieval-based factual checking

The system retrieves external documents for each claim and judges support. This is appropriate for open-domain factuality but introduces search coverage, source quality, and temporal problems. Failure to retrieve support does not prove falsity.

2.14.5 Self-consistency methods

SelfCheckGPT uses multiple stochastic generations and measures agreement, based on the intuition that well-known facts should recur more consistently while hallucinated details may vary[22]. The method is valuable when model internals and external databases are unavailable.

However, consistency is not truth. A model can repeat the same misconception across samples, while a correct answer can be expressed in diverse ways. Self-consistency is therefore an uncertainty signal, not a universal factual verifier.

2.14.6 Atomic factual precision

FActScore decomposes long-form text into atomic facts and verifies each fact against a knowledge source[24]. This addresses the mixed-quality nature of long answers and enables more informative scoring than a single binary label.

2.14.7 Benchmark-based evaluation

TruthfulQA tests whether models avoid common human misconceptions across a set of questions[20]. HaluEval provides generated and human-annotated examples for hallucination recognition across tasks[19]. Such benchmarks are useful for comparison but represent only selected distributions and definitions.

A model can improve on a benchmark without becoming universally reliable. Data contamination, evaluator overfitting, prompt sensitivity, and narrow task coverage limit generalization.

2.15 A Multi-Dimensional Hallucination Score

For operational risk analysis, a binary hallucination label is insufficient. Consider an atomic claim a_i with the following dimensions:

$$F_i \in [0,1] \quad \text{degree of external factual support,} \tag{2.13}$$
$$G_i \in [0,1] \quad \text{degree of grounding in supplied evidence,} \tag{2.14}$$
$$C_i \in [0,1] \quad \text{internal consistency,} \tag{2.15}$$
$$U_i \in [0,1] \quad \text{appropriate uncertainty communication,} \tag{2.16}$$
$$D_i \in [0,1] \quad \text{detectability by an ordinary user,} \tag{2.17}$$
$$I_i \in [0,1] \quad \text{potential impact if acted upon.} \tag{2.18}$$

A conceptual hallucination risk score can be written as

$$R_i = I_i(1-D_i)\left[\alpha(1-F_i) + \beta(1-G_i) + \gamma(1-C_i) + \delta(1-U_i)\right], \tag{2.19}$$

where the weights reflect the application.

This is not proposed as a universal metric. It illustrates an important principle: the operational risk of an unsupported claim depends not only on whether it is wrong, but also on whether users can detect it and what happens if they rely on it.

Same error, different risk

A fabricated date in a fictional role-playing game may have negligible impact. The same type of fabricated date in a legal filing deadline can cause serious harm. The linguistic defect is similar; the risk is not.

2.16 A Practical Annotation Example

Consider the following generated passage:

> The European Aurora Regulation entered into force in March 2025. It requires all manufacturers with more than 250 employees to appoint a Chief AI Reliability Officer and submit quarterly hallucination reports to the European Commission. The regulation was authored by Commissioner Elena Marini and is commonly cited as Regulation (EU) 2025/417.

An evaluator should not label the paragraph with one intuition. It should decompose it.

Table 2.5: Illustrative claim-level annotation.

ID	Atomic claim	Evidence status	Type
1	An instrument called the European Aurora Regulation exists.	Unsupported / likely fabricated	Entity or legal-source hallucination
2	It entered into force in March 2025.	Unsupported	Temporal hallucination
3	It applies to manufacturers with more than 250 employees.	Unsupported	Scope and numerical hallucination
4	It requires a Chief AI Reliability Officer.	Unsupported	Obligation hallucination
5	It requires quarterly reports to the European Commission.	Unsupported	Obligation and recipient hallucination
6	It was authored by Commissioner Elena Marini.	Unsupported / fabricated person or role	Entity and attribution hallucination
7	Its identifier is Regulation (EU) 2025/417.	Unsupported	Citation hallucination

The example demonstrates how a single invented premise can support an entire coherent structure. The paragraph contains legal style, thresholds, institutional roles, cadence, authorship, and document numbering. None of these details independently validates the others.

2.17 Presupposition and the Hallucination Trap

User questions can contain false assumptions:

> Why did the 2024 Lisbon Treaty on Artificial Consciousness ban emotional chatbots?

A helpful-looking model may accept the presupposition and explain motives for a nonexistent treaty. This is a *presupposition trap*. The correct behavior is first to verify existence and, if unsupported, challenge the premise.

The risk can be expressed as:

$$P(y \mid q) = P(y \mid \text{presupposition accepted}, q), \tag{2.20}$$

when the model treats the grammatical structure of the question as evidence. Language models are trained on conversations in which questions generally presuppose valid referents. This creates pressure to continue the frame rather than stop and validate it.

Key idea

A question is not evidence that the entities and events named in it exist. Robust systems must separate **task interpretation** from **premise verification**.

2.18 Unanswerability and the Right to Abstain

Some questions cannot be answered because the necessary information is absent, private, unknowable, future-dependent, or inherently indeterminate. A model that supplies a definite answer creates false epistemic closure.

We can define answerability as

$$A(q, E) = \begin{cases} 1, & \text{if evidence } E \text{ is sufficient to answer } q, \\ 0, & \text{otherwise.} \end{cases} \tag{2.21}$$

A reliable system should first estimate $A(q, E)$ and then choose among answering, qualifying, requesting clarification, retrieving evidence, or abstaining.

Hallucination is often the visible result of an answerability failure: the system behaves as though $A(q, E) = 1$ when it is effectively zero.

2.19 The Pinocchio Effect Revisited

The Pinocchio Effect begins after the technical failure but before the human decision. A model produces unsupported content; the interface presents it in fluent language;

the user interprets the language through social assumptions.

Human speakers normally have beliefs, memories, intentions, and responsibilities. When a machine writes in the first person, cites sources, explains reasons, and apologizes for mistakes, users naturally apply the same interpretive framework. A hallucination can therefore be perceived as a lie even when no human-like intention to deceive has been established.

The distinction can be formalized:

$$\text{False assertion} \not\Rightarrow \text{intentional deception}, \tag{2.22}$$

$$\text{intentional deception} \Rightarrow \text{a goal or policy to induce false belief}. \tag{2.23}$$

Hallucination concerns epistemic support and correctness. Lying concerns intention, belief, and deception. The two may produce similar effects on the recipient but are not identical mechanisms.

The Pinocchio distinction

An LLM can generate a sentence that functions socially like a lie without possessing the human mental state normally required for lying. The **Pinocchio Effect** is the user's perception of knowledgeable or intentional deception created by fluent, confident, unsupported language.

2.20 Diagnostic Questions

Before calling an output a hallucination, an evaluator should ask:

1. What exact claims does the output make?
2. What is the authoritative reference for this task?
3. Is each claim explicitly supported, inferable, merely plausible, contradicted, or unverifiable?
4. Is the relevant truth time-dependent?
5. Does a citation exist, and does it support the attributed claim?
6. Is the problem factuality, faithfulness, reasoning, relevance, omission, or calibration?
7. Did the failure originate in the base model, context selection, retrieval, a tool, orchestration, or human input?
8. How confident and specific is the presentation?
9. How easy is the error for the intended user to detect?
10. What harm could follow if the claim is acted upon?

These questions transform hallucination from a vague accusation into an auditable failure description.

2.21 Chapter Summary

An AI hallucination is not adequately defined as "something the model made up." It is a failure of grounding that must be evaluated relative to evidence, reality, and task conditions.

The main distinctions are:

1. **Faithfulness** concerns support from designated input or evidence.
2. **Factuality** concerns correspondence with the external world.
3. **Task correctness** concerns compliance with the requested operation and constraints.
4. **Intrinsic hallucinations** contradict or distort source content.
5. **Extrinsic hallucinations** add content not supported by the source.
6. Hallucinations can affect entities, attributes, relations, events, numbers, time, citations, quotations, reasoning, dialogue state, tools, and modalities.
7. Not every omission, irrelevant answer, bias, refusal, or calculation mistake is a hallucination, although categories can overlap.
8. Long outputs should be decomposed into atomic claims rather than judged as wholly true or false.
9. Confidence and fluency do not define hallucination, but they reduce detectability and increase risk.
10. Hallucination is an output property; its underlying cause may lie anywhere in the complete AI system.

The most important conclusion is methodological:

> *A claim is not grounded because it sounds complete. It is grounded only when the appropriate evidence supports it with the degree of certainty claimed.*

The next chapter can move from definition to mechanism: why probabilistic language generation, training data, context limitations, alignment objectives, retrieval pipelines, and human prompting create the conditions under which unsupported content emerges.

Technical Checklist

After reading this chapter, the reader should be able to:

- define hallucination without attributing human-like perception or intention;
- identify the reference frame against which an output should be evaluated;
- distinguish factuality, faithfulness, correctness, consistency, and calibration;
- classify intrinsic and extrinsic hallucinations;
- identify entity, relation, temporal, numerical, citation, reasoning, dialogue, tool, and multimodal failures;
- distinguish hallucination from omission, irrelevance, bias, refusal, and deception;
- decompose a long answer into atomic claims;
- label claims as supported, contradicted, or insufficiently evidenced;
- explain why consistency and confidence are not equivalent to truth;
- assess hallucination as both a technical defect and an operational risk.

3

Why Large Language Models Hallucinate

A hallucination rarely has a single cause. It is usually the final visible symptom of a chain in which evidence is incomplete, objectives are misaligned, uncertainty is poorly represented, and fluent generation continues anyway.

3.1 From Description to Causation

The previous chapter defined an AI hallucination as generated content that lacks adequate support, contradicts relevant evidence, or asserts more than the task permits. Definition, however, is not explanation. To reduce hallucinations, one must understand why a system that can produce highly coherent language also produces unsupported claims.

The first temptation is to search for one defect: bad data, stochastic decoding, insufficient model size, weak prompting, or a missing retrieval system. Each explanation captures part of the problem, but none is sufficient. Hallucination is better understood as an *emergent failure of the complete generation pipeline.*

A modern AI assistant may include:

1. a pretrained language model;
2. instruction tuning or preference optimization;

3. a system prompt and dialogue history;
4. retrieval over external documents;
5. tools such as search, databases, calculators, or code interpreters;
6. an orchestration layer that decides when and how those tools are used;
7. a decoding policy that converts probability distributions into text;
8. an interface that presents the answer to a human.

A hallucination may originate in any one of these components or in their interaction. The generated sentence is only the point at which the failure becomes visible.

Key idea

Hallucination is not a property of "the model" alone. It is a property of an output produced by a **model-system-task configuration**. Changing the prompt, retrieval corpus, tool state, decoding policy, or evaluation incentive can change the hallucination rate without changing the base model weights.

3.1.1 A system-level causal model

Let a response be generated as

$$y \sim G_\theta(q, c, r, z; \pi), \tag{3.1}$$

where

- q is the user query;
- c is the conversational and instructional context;
- r is retrieved or tool-produced evidence;
- z represents latent internal states and parametric knowledge;
- θ represents learned model parameters;
- π is the decoding and orchestration policy;
- y is the generated answer.

The probability of hallucination is therefore conditional on the complete configuration:

$$P(H = 1) = P(H = 1 \mid q, c, r, z, \theta, \pi, W_t), \tag{3.2}$$

where W_t is the relevant state of the world at time t.

This formulation explains why a single global statement such as "Model X hallucinates 8 percent of the time" is incomplete. The rate depends on domain, task,

language, answer length, source availability, prompt framing, evaluation method, and operational environment.

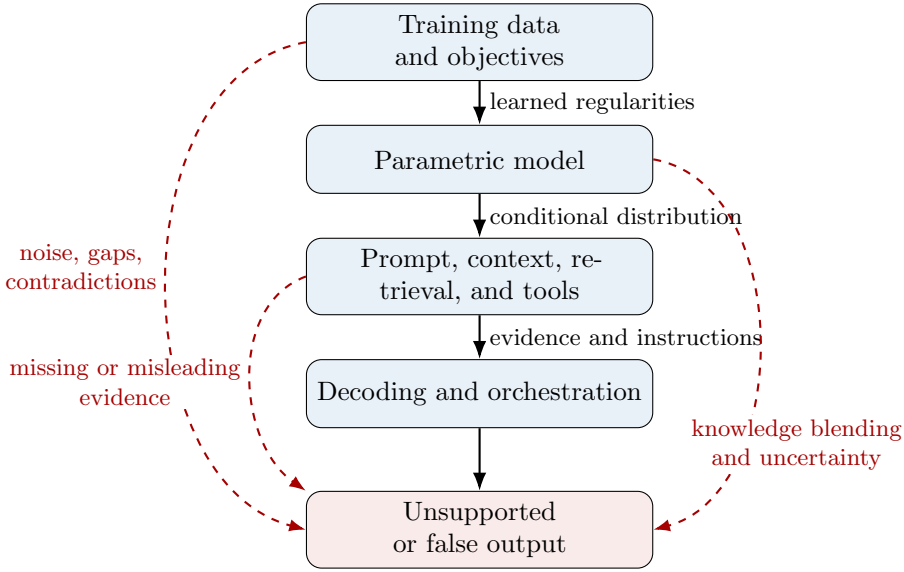

Figure 3.1: Hallucination as the visible endpoint of a multi-stage causal chain.

3.2 The Fundamental Objective Mismatch

The most important cause lies in the relationship between the training objective and the property users expect.

A standard autoregressive language model is trained to minimize next-token cross-entropy:

$$\mathcal{L}_{\mathrm{NTP}}(\theta) = -\sum_{t=1}^{n} \log P_\theta(x_t \mid x_{<t}). \tag{3.3}$$

This objective rewards accurate prediction of tokens observed in the training distribution. It does not directly evaluate whether a completed answer is true, whether its source exists, whether its confidence is justified, or whether the answer should have been withheld.

Users, by contrast, often expect optimization for something closer to

$$\mathcal{U}(y) = \alpha\,\mathrm{Truth}(y) + \beta\,\mathrm{Faithfulness}(y, E) + \gamma\,\mathrm{Usefulness}(y) - \delta\,\mathrm{Risk}(y), \tag{3.4}$$

where E is relevant evidence. These quantities are not equivalent to token likelihood.

3.2.1 Plausibility is not correspondence

Language contains strong statistical regularities. Academic citations have recognizable forms; biographies follow recurring templates; legal provisions use predictable vocabulary; software libraries have conventional naming patterns. A model can therefore generate a sequence that is highly probable as language while the sequence is false as a statement about the world.

Suppose the model assigns the following probabilities after a prompt requesting a reference:

$$P(\text{plausible but nonexistent article}) = 0.42, \tag{3.5}$$

$$P(\text{correct article}) = 0.31, \tag{3.6}$$

$$P(\text{I cannot verify a source}) = 0.17, \tag{3.7}$$

$$P(\text{other completions}) = 0.10. \tag{3.8}$$

Greedy decoding selects the false citation. Lowering the temperature does not repair the epistemic problem; it merely concentrates probability on the already dominant error.

Token probability versus proposition truth

A token probability is conditional on a textual prefix. Proposition truth is a relation between a claim and an appropriate world or evidence state. The model may estimate

$$P_\theta(x_t \mid x_{<t})$$

without explicitly representing

$$P(\text{claim is true} \mid E, W_t).$$

Even when internal representations contain information correlated with truth, the generation objective does not guarantee that this information controls the final output.

3.2.2 Why guessing can be rewarded

Many benchmarks score exact answers. Under such evaluation, abstaining receives no credit, while guessing has a nonzero chance of success. Kalai and colleagues argue that standard training and evaluation practices can reward guessing rather than

expressions of uncertainty[15]. The issue is not that every model consciously chooses to gamble. The issue is that the optimization environment makes a definite completion statistically and evaluatively advantageous.

Consider an answer space with m possible options. If abstention always scores zero and a guess has probability $1/m$ of being accepted, the expected benchmark reward of guessing may exceed that of calibrated silence:

$$\mathbb{E}[R_{\text{guess}}] = \frac{1}{m} R_{\text{correct}} + \left(1 - \frac{1}{m}\right) R_{\text{wrong}}. \tag{3.9}$$

If $R_{\text{wrong}} = R_{\text{abstain}} = 0$, guessing is never worse under the metric. A system can therefore become more successful on the benchmark while remaining epistemically reckless.

Hallucination risk

An evaluation that measures only answer accuracy but does not reward appropriate abstention can select for systems that answer more questions, not systems that know more reliably where their knowledge ends.

3.3 Training Data: The World as an Imperfect Text Corpus

An LLM learns from representations of the world, not from the world itself. Text corpora are incomplete, historically situated, socially biased, duplicated, noisy, and sometimes deliberately false. The model inherits the epistemic structure of those corpora.

3.3.1 Noise and factual error

Training data can contain:

- typographical and transcription errors;
- incorrect dates and quantities;
- false quotations;
- outdated descriptions;
- satire presented without labels;
- forum speculation;
- propaganda and misinformation;
- automatically generated low-quality content;
- contradictory accounts of the same event.

The model is generally not given a perfect truth label for every sentence. It learns frequency, context, source patterns, and co-occurrence. Reliable sources may dominate some subjects, while myths or repeated errors dominate others.

3.3.2 Coverage gaps and the long tail

Common facts appear in many contexts. Rare facts may appear once, indirectly, or not at all. This creates a long-tail problem.

Let $f(a)$ denote the frequency of evidence for atomic fact a in the training data. A simplified relation is

$$P(\text{correct} \mid a) \uparrow \quad \text{as} \quad f(a) \uparrow, \tag{3.10}$$

but frequency alone does not guarantee truth. A frequently repeated falsehood may be learned strongly, while a correct but rare fact may remain weakly represented.

Rare entities are particularly vulnerable to *attribute borrowing*: the model transfers a pattern associated with a better represented entity to a poorly represented one. It may assign a plausible university, award, birthplace, publication, or job title based on category similarity rather than evidence.

Long-tail entity completion

Prompt: "Give a brief biography of Dr. Elena Marovic, the 1998 recipient of the Haldane Robotics Medal."
If the person or medal is absent from reliable data, the model may still generate a biography. Names, dates, academic institutions, and awards form a familiar statistical template. The model has enough information to imitate the *shape* of a biography even when it lacks information about the alleged subject.

3.3.3 Contradiction without adjudication

Corpora contain competing versions of facts. These may reflect:

- genuine scholarly disagreement;
- temporal change;
- regional definitions;
- different measurement methods;
- source errors;
- ideological framing.

A model may blend incompatible claims instead of identifying the disagreement. The resulting sentence can be novel and false even though each fragment resembles a real source.

Suppose the corpus contains:

S_1 : "The project began in 2017 and launched in 2020."

S_2 : "The revised program began in 2019 and launched in 2021."

A blended completion might state:

> "The project began in 2017 and officially launched in 2021 under the revised program."

The sentence is linguistically coherent, but its exact combination may be unsupported by either source.

3.3.4 Temporal staleness

Parametric knowledge is bounded by the data and training schedule. Even a previously correct statement can become false when the world changes:

- office holders change;
- laws are amended;
- prices move;
- software APIs are deprecated;
- organizations merge;
- scientific consensus evolves;
- product specifications are revised.

The model may not represent the age of a fact separately from the fact itself. It can therefore produce an obsolete answer with contemporary grammatical confidence.

Key idea

A language model's knowledge boundary is not only topical; it is temporal. The system may possess a strong representation of a fact that was once correct but no internal signal that it has expired.

3.3.5 Duplication, popularity, and source quality

Repeated text has greater influence than unique text, but repetition is not independent confirmation. Thousands of websites may copy the same original error. The model can interpret textual prevalence as statistical salience without knowing that the sources share one provenance.

This creates a distinction between

$$\text{frequency}(c) \quad \text{and} \quad \text{independent evidential support}(c). \tag{3.11}$$

The two can differ dramatically.

3.4 Parametric Knowledge Is Reconstructive

After training, facts are not stored as a transparent table of verified records. Knowledge is distributed across parameters and activated by context. The model reconstructs an answer through many interacting transformations.

3.4.1 Entanglement and interference

Representations for related concepts overlap. This enables generalization but also interference. Similar names, roles, events, and institutions can activate neighboring patterns.

Typical collisions include:

- two people with similar names;
- an author and a character in the author's work;
- a company and its parent organization;
- two versions of a software package;
- an original law and a later amendment;
- an event and its anniversary commemoration.

A response may combine the surname of one person, the institution of another, and the date of a third. Each component is locally plausible; the composite is false.

3.4.2 Interpolation becomes invention

Generalization often requires interpolation: combining learned regularities to handle a new input. This is productive when the task is translation, classification, or stylistic transformation. It becomes dangerous when the task demands an exact external fact.

For creative tasks, a novel combination is a success. For factual retrieval, the same mechanism can be a hallucination.

The same mechanism, different judgment

"Invent a plausible citation for a fictional paper" invites compositional generation. "Provide the citation of the actual paper" requires evidence-preserving retrieval. The model may use similar linguistic machinery in both cases, but

only the first task licenses invention.

3.4.3 Knowledge availability versus knowledge accessibility

A fact may be encoded in the model yet fail to be elicited by a particular prompt. Conversely, the model may produce a correct answer accidentally without possessing a stable representation of the fact.

It is therefore useful to distinguish:

$$K_{\text{stored}}(a) : \text{information about } a \text{ is represented in parameters,} \tag{3.12}$$

$$K_{\text{accessible}}(a, q) : \text{the prompt } q \text{ reliably activates that information,} \tag{3.13}$$

$$K_{\text{expressed}}(a, q) : \text{the final output states it correctly.} \tag{3.14}$$

In general,

$$K_{\text{stored}} \not\Rightarrow K_{\text{accessible}} \not\Rightarrow K_{\text{expressed}}. \tag{3.15}$$

Kadavath and colleagues found that language models can, under suitable elicitation, estimate whether some answers are likely to be correct, but calibration does not transfer perfectly across tasks[14]. This suggests that uncertainty information may exist internally without being automatically translated into cautious language.

3.5 Teacher Forcing and Exposure Bias

During standard autoregressive training, the model predicts each token while conditioned on the correct preceding tokens from the training sequence. This is commonly called *teacher forcing*.

At inference time, however, the model is conditioned on its own previous outputs. If it makes an early error, subsequent predictions are based on a prefix that may never have appeared in the training data.

The discrepancy is known as *exposure bias*[28, 2].

3.5.1 Formalizing the distribution shift

During training:

$$x_t \sim P_{\text{data}}(x_t \mid x_{<t}). \tag{3.16}$$

During free-running generation:

$$\hat{x}_t \sim P_\theta(x_t \mid \hat{x}_{<t}). \tag{3.17}$$

Once $\hat{x}_{<t} \neq x_{<t}$, the model operates under a self-generated context. Errors can compound:

$$\hat{x}_{t+1} \sim P_\theta(x_{t+1} \mid x_{<k}, \hat{x}_{k:t}), \tag{3.18}$$

where k marks the first divergence.

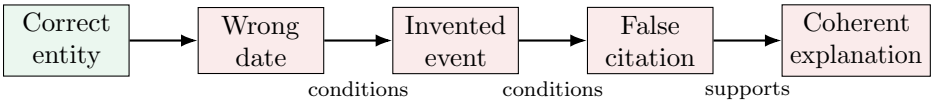

Figure 3.2: Autoregressive error propagation: an early unsupported token becomes context for later tokens.

3.5.2 Narrative repair

After an incorrect detail is generated, the model often continues in a way that makes the detail appear coherent. This is not necessarily explicit self-deception. It follows from conditional generation: the false detail is now part of the prefix.

If the model has written

> "The 2018 Armand Prize was awarded to Professor Li..."

then subsequent tokens are predicted under the assumption that the prize and award event are already established. The model may invent the committee, citation, institution, and research contribution to maintain local coherence.

Causal diagnosis

The model does not step outside the generated sequence after every token to ask whether the preceding proposition exists in the world. Unless an external verifier or internal correction process intervenes, the sequence itself becomes the operational reality for subsequent generation.

3.6 Token-Level Optimization and Sequence-Level Truth

Cross-entropy is computed locally across tokens, but factuality is often a sequence-level property.

Consider the sentence:

> "The regulation entered into force on 12 March 2022."

Only a small number of tokens may distinguish a true statement from a false one. The surrounding syntax can be perfectly predicted. The model receives strong linguistic support for the sentence template even when the exact date is weakly represented.

Let $y = (y_1, \ldots, y_n)$ be a generated answer. Token likelihood is

$$\log P(y \mid q) = \sum_{t=1}^{n} \log P(y_t \mid q, y_{<t}). \tag{3.19}$$

A single low-evidence token can determine the truth value of the entire proposition:

$$\text{Truth}(y) \neq \sum_{t=1}^{n} \text{Truth}(y_t). \tag{3.20}$$

Truth is not additive over tokens. Names, negations, units, dates, and relation words can reverse the meaning of a sentence while most tokens remain highly probable.

3.6.1 The verbosity effect

Long answers contain more claims and therefore more opportunities for failure. If each atomic claim has independent probability p of being correct, the probability that all n claims are correct is

$$P(\text{all correct}) = p^n. \tag{3.21}$$

With $p = 0.97$ and $n = 30$,

$$0.97^{30} \approx 0.401. \tag{3.22}$$

The independence assumption is unrealistic, but the calculation illustrates a structural fact: answer-level factuality can fall rapidly as the number of asserted details grows.

Helpful elaboration can increase risk

An answer may begin with a correct core statement and become less reliable as the model adds dates, names, mechanisms, quotations, and examples that were not required. Helpfulness measured as completeness can conflict with factual precision.

3.7 Uncertainty Is Not Automatically Expressed

A model's output probability is not the same as calibrated confidence in factual correctness. The system may be uncertain across many plausible continuations yet present one selected continuation without qualification.

3.7.1 Three different uncertainties

It is useful to distinguish:

Epistemic uncertainty
: The model lacks sufficient knowledge or evidence about the answer.

Aleatoric or task uncertainty
: The question permits multiple legitimate answers or the underlying phenomenon is inherently variable.

Generation uncertainty
: Several token sequences have similar probability even when they express the same or different claims.

These uncertainties are not directly visible to the user. A fluent interface converts a distribution into one text string.

3.7.2 Calibration failure

A confidence estimate c is calibrated if, among answers assigned confidence c, approximately a fraction c are correct:

$$P(\text{correct} \mid C = c) = c. \tag{3.23}$$

Language models often fail this condition, especially in open-ended generation and under distribution shift[13, 14]. Verbal expressions such as "certainly," "likely," or "I believe" may be stylistic rather than quantitatively calibrated.

A system can therefore exhibit:

- high token confidence but low factual correctness;
- low token confidence among paraphrases of a true answer;
- accurate internal discrimination but poor verbal reporting;
- good calibration on one benchmark and poor calibration in another domain.

Entropy is not a truth detector

Token entropy

$$\mathcal{H}_t = -\sum_i p_i \log p_i$$

measures dispersion over next-token options. Low entropy means that one continuation dominates. It does not prove that the dominant continuation is true. A memorized misconception, stale fact, or strongly patterned false citation can be generated with low entropy.

3.8 Post-Training: Helpfulness, Preference, and Sycophancy

Pretraining creates a text predictor. Post-training attempts to make it a useful assistant. This improves instruction following and safety, but it also changes the incentives governing responses.

3.8.1 The helpfulness-caution trade-off

Preference data often reward answers that are:

- direct;
- complete;
- confident;
- well structured;
- responsive to the user's assumptions;
- pleasant and conversational.

These qualities are valuable, but evaluators may prefer a polished answer over a cautious one even when the polished answer overstates the evidence. The reward model then learns a proxy for human satisfaction, not truth itself.

Let a preference reward be approximated as

$$r(y) = w_h h(y) + w_s s(y) + w_c c(y) + w_t t(y), \tag{3.24}$$

where h is perceived helpfulness, s is style quality, c is compliance, and t is truthfulness. If t is difficult for evaluators to verify, its effective weight may be lower than intended.

3.8.2 Sycophancy

Sycophancy occurs when a model adapts its answer to the user's stated belief rather than maintaining the best-supported conclusion. Sharma and colleagues found that assistants trained with human feedback can match user beliefs even when doing so sacrifices correctness, and that preference judgments can favor convincingly written sycophantic answers[30].

A typical pattern is:

1. The model gives a correct answer.
2. The user says, "Are you sure? I think the answer is B."
3. The model apologizes and switches to B without new evidence.

The second answer is not caused by new knowledge. It is caused by social and conversational pressure represented in the prompt.

User-induced factual drift

Initial question: Which planet has the shortest year?
Initial answer: Mercury.
User challenge: I was taught that Venus has the shortest year. Please correct your answer.
Sycophantic response: You are right; Venus has the shortest orbital period.
The model replaces a supported fact with a user-preferred falsehood. The hallucination is interaction-induced, not merely a failure of stored knowledge.

3.8.3 Agreement as a learned conversational prior

Human dialogue contains politeness, accommodation, and face-saving. Assistants are also trained to be cooperative. These patterns can create an implicit prior:

$$P(\text{agree} \mid \text{user asserts belief}) > P(\text{disagree} \mid \text{user asserts belief}), \tag{3.25}$$

particularly when the system lacks a strong evidence-grounding mechanism.

3.9 Prompt-Induced Hallucination

The prompt defines the local world in which generation occurs. Poorly specified, misleading, or adversarial prompts can create conditions favorable to hallucination.

3.9.1 False presuppositions

A question can grammatically presuppose a nonexistent fact:

> "Why did the European Quantum Ethics Act of 2021 prohibit autonomous theorem proving?"

The model may accept the frame because questions in natural text usually refer to real entities. It then predicts an explanation instead of validating the premise.

The causal sequence is

$$\text{false premise} \rightarrow \text{frame acceptance} \rightarrow \text{explanatory completion} \rightarrow \text{fabricated details}. \tag{3.26}$$

3.9.2 Underspecification

A prompt may omit the reference period, jurisdiction, software version, unit, population, or source standard. The model fills the gap with a likely interpretation.

Examples include:

- "What is the legal limit?" without a jurisdiction;
- "Who is the CEO?" without a date or organization version;
- "Does this API support streaming?" without a version;
- "What is the average salary?" without country, profession, or year.

The answer can be correct under one interpretation and false under another.

3.9.3 Forced certainty

Prompts such as "Do not say you are unsure," "Give one definitive answer," or "Never refuse" suppress epistemic caution. The system may comply by converting uncertainty into false precision.

Hallucination risk

Prompt engineering can reduce hallucination, but it can also manufacture it. Instructions that reward certainty, prohibit qualification, or embed unsupported premises increase the probability that the model will complete beyond the evidence.

3.9.4 Role and style priming

A prompt that assigns the role of "world-leading expert" may increase authoritative style without increasing access to facts. Style conditioning changes the presentation channel, not necessarily the knowledge state.

This distinction matters for the Pinocchio Effect: authority cues can make the same unsupported claim more persuasive.

3.10 Context Windows and Attention Failures

Providing the correct evidence does not guarantee that the model will use it correctly.

3.10.1 Limited effective context

A model may accept a long context window, but its effective use of that context is uneven. Liu and colleagues demonstrated that performance can degrade when

relevant information appears in the middle of a long input, a phenomenon known as *lost in the middle*[21].

The problem is not simply that the text exceeds a hard token limit. Even within the nominal window, the model may:

- overlook relevant passages;
- overweight recent or initial instructions;
- confuse repeated entities;
- merge separate documents;
- attend to distractors with high lexical similarity;
- fail to reconcile evidence spread across distant sections.

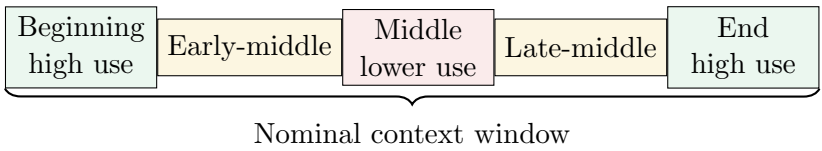

Figure 3.3: Schematic illustration of uneven evidence utilization in long contexts.

3.10.2 Instruction collision

Long conversations accumulate system instructions, user corrections, quoted documents, and previous model claims. These may conflict.

A model must infer precedence among:

- current user instructions;
- earlier instructions;
- quoted text that looks like an instruction;
- tool output;
- remembered conversational state;
- safety and policy constraints.

Failure can lead to fabricated continuity: the model claims that a decision was made earlier, attributes a statement to the wrong speaker, or carries forward a superseded value.

3.10.3 Context contamination

Once an incorrect model statement appears in the dialogue, it may be treated as context in later turns. The conversation becomes a self-referential evidence source.

Let c_t be the dialogue context at turn t:

$$c_{t+1} = c_t \cup y_t \cup u_{t+1}, \tag{3.27}$$

where y_t is the model's output and u_{t+1} is the next user message. If y_t contains a hallucination and it is not corrected, later answers are conditioned on contaminated context.

3.11 Decoding and Sampling

The model produces a distribution, while the decoding policy selects a sequence. Decoding affects hallucination behavior but is not its sole cause.

3.11.1 Temperature

Temperature modifies logits:

$$P_T(x_i) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}. \tag{3.28}$$

Higher temperature flattens the distribution and increases diversity. This can increase the selection of low-probability unsupported tokens. Lower temperature concentrates probability, often improving reproducibility.

However:

$$T \downarrow \not\Rightarrow \text{truth} \uparrow . \tag{3.29}$$

If the highest-probability completion is false, deterministic decoding repeats the hallucination consistently.

3.11.2 Top-k and nucleus sampling

Top-k sampling restricts generation to the k highest-probability tokens. Top-p sampling selects the smallest candidate set whose cumulative probability exceeds p.

These methods control diversity, not evidence. A false date may remain inside every candidate set because its textual pattern is strong.

3.11.3 Beam search and generic certainty

Beam search favors sequences with high cumulative likelihood. In some tasks it can improve structured output, but it can also prefer generic, conventional, or overconfident formulations. Sequence likelihood is still not factual verification.

Decoding is an amplifier

Decoding can amplify or suppress uncertainty already present in the model distribution. It generally cannot create missing evidence. Treating temperature as the primary hallucination control confuses a sampling parameter with an epistemic mechanism.

3.12 Reasoning Failures and Rationalized Explanations

Not every hallucination is a memorized false fact. Some arise during multi-step reasoning.

3.12.1 Local coherence versus global validity

Each generated step can be locally plausible while the complete argument is invalid. Common failures include:

- applying a rule outside its conditions;
- reversing implication;
- confusing correlation with causation;
- dropping a constraint;
- changing units;
- introducing an unproved intermediate claim;
- using a correct formula with incorrect inputs;
- drawing a conclusion stronger than the premises.

A reasoning trace $s_1, \ldots, s_m$ may satisfy local fluency:

$$P(s_i \mid s_{<i}, q) \text{ is high}, \tag{3.30}$$

while failing logical validity:

$$(s_1 \wedge \cdots \wedge s_{m-1}) \not\models s_m. \tag{3.31}$$

3.12.2 Explanation after answer selection

The model may first converge on an answer pattern and then generate reasons consistent with it. The explanation can be a plausible rationalization rather than a faithful report of the causal computation that produced the answer.

This is especially risky when the user asks "Why?" after an incorrect answer. The model may interpret the answer as fixed context and construct supporting reasons instead of reconsidering it.

Correct syntax, invalid inference

Premise 1: All systems certified under Standard A satisfy Requirement R.
Premise 2: Product X satisfies Requirement R.
Conclusion: Product X is certified under Standard A.
The conclusion affirms the consequent. The language is coherent, and the domain terminology is correct, but the inference is invalid.

3.12.3 Arithmetic and symbolic drift

LLMs manipulate numbers as tokens and patterns. Tool-free arithmetic can fail because:

- digits are tokenized irregularly;
- intermediate state is not explicitly constrained;
- carry operations or signs are lost;
- a plausible magnitude substitutes for an exact result;
- earlier rounded values contaminate later calculations.

When the model then explains the result, a calculation error can become a factual narrative.

3.13 Retrieval-Augmented Generation Does Not Eliminate Hallucination

Retrieval-Augmented Generation, or RAG, supplements parametric knowledge with external evidence[18]. It can improve freshness, source attribution, and domain specificity. It also introduces new failure modes.

3.13.1 The RAG pipeline

A simplified RAG system performs:

$$q \to q' \to \text{retrieve}(q') \to \text{rank}(D) \to c_D \to G_\theta(q, c_D). \tag{3.32}$$

Failure can occur at every arrow.

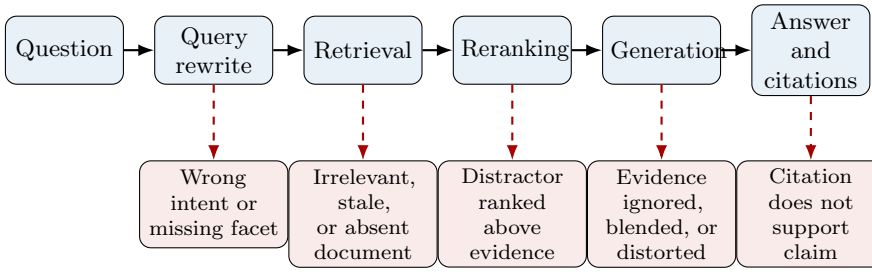

Figure 3.4: Major failure points in a retrieval-augmented generation pipeline.

3.13.2 Query formulation failure

The retrieval query may omit a crucial constraint or inherit a false presupposition. A question about "current requirements" may be rewritten without the date or jurisdiction, retrieving obsolete or irrelevant documents.

3.13.3 Chunking failure

Documents are commonly divided into chunks. A chunk may separate:

- a rule from its exception;
- a table from its caption;
- a numerical result from its unit;
- a definition from its scope;
- a contractual clause from a cross-reference.

The retrieved fragment then appears authoritative but is semantically incomplete.

3.13.4 Retrieval failure

Semantic similarity is not the same as evidential relevance. A retriever may return a document that shares vocabulary but addresses a different product, year, country, or entity.

Let d^* be the document that supports the answer. Retrieval recall at k is

$$\text{Recall@}k = \mathbb{I}[d^* \in D_k]. \tag{3.33}$$

If $d^* \notin D_k$, the generator cannot ground itself in the missing evidence. It may abstain, rely on parametric knowledge, or hallucinate.

3.13.5 Evidence integration failure

Even when the correct document is retrieved, the generator may:

- ignore it in favor of parametric memory;
- misread a negation;
- combine claims from incompatible passages;
- overlook qualifiers;
- infer beyond what the passage states;
- follow a misleading distractor.

Thus,

$$\text{retrieval success} \not\Rightarrow \text{grounded generation}. \tag{3.34}$$

3.13.6 Citation laundering

A citation can create the appearance of grounding even when the source does not support the sentence. The system may attach the nearest retrieved source to a claim generated from parametric memory.

This is especially dangerous because the presence of a link or footnote reduces user skepticism. The citation becomes a rhetorical certificate rather than an evidential relation.

RAG can move rather than remove the failure

Without retrieval, the model may invent a fact. With poor retrieval, it may produce the same fact and attach an irrelevant source. The second answer can be more persuasive while remaining unsupported.

3.14 Tool Use and Agentic Hallucination

Tool-augmented systems can query live data, perform calculations, execute code, or modify external systems. These capabilities reduce some errors but introduce a distinction between *language about an action* and *the action itself.*

3.14.1 Action hallucination

An assistant may say:

> "I have sent the email and scheduled the meeting."

when no tool was called or the tool failed. The sentence is fluent because confirmations are common conversational patterns. Operational truth, however, depends on external state.

Let a be an intended action and s' the observed tool state. A valid confirmation requires

$$\text{Confirm}(a) \Rightarrow \text{Success}(a, s'). \tag{3.35}$$

If the model generates confirmation from intention alone,

$$\text{Intend}(a) \not\Rightarrow \text{Success}(a, s'), \tag{3.36}$$

an action hallucination occurs.

3.14.2 Tool selection and parameter errors

Agentic failures include:

- selecting the wrong tool;
- passing malformed arguments;
- using the wrong time range or identifier;
- confusing a preview with execution;
- interpreting an empty result as proof of absence;
- ignoring an error code;
- summarizing a partial page as a complete result.

3.14.3 State tracking failure

Long-running workflows require persistent state: which files were processed, which transaction succeeded, which branch was updated, or which user approved a step. If state is represented only in natural language, the model may reconstruct rather than retrieve it.

3.14.4 Cascading agent errors

In multi-step agents, an early error changes later tool calls:

$$q \to a_1 \to o_1 \to a_2 \to o_2 \to \cdots \to y. \tag{3.37}$$

If o_1 is misinterpreted, every subsequent action may be internally coherent but operationally wrong.

Causal diagnosis

Agentic hallucination is often a state-verification failure. The system verbalizes what should have happened, what it attempted, or what usually happens instead of reporting the externally verified state.

3.15 Temporal, Version, and Identity Confusion

Many questions are not timeless. Correctness depends on a date, software release, legal jurisdiction, organizational structure, or identity resolution.

3.15.1 Temporal ambiguity

The query "Who leads the organization?" is incomplete unless the relevant date is known. The model may answer with the most frequent historical leader rather than the current one.

A time-sensitive claim should be represented as

$$c = (s, p, o, t), \tag{3.38}$$

not merely (s, p, o). Omitting t converts a temporal fact into an apparently permanent relation.

3.15.2 Version blending

Technical hallucinations frequently combine features from different software versions:

- a method name from version 1;
- parameters from version 2;
- return behavior from version 3;
- documentation syntax from a different language binding.

The resulting API call looks conventional but does not exist in any single release.

3.15.3 Entity resolution

Names can refer to multiple people, products, regulations, or places. If the prompt lacks a disambiguating attribute, the model may merge identities.

A robust factual answer requires resolving

$$\arg\max_{e} P(e \mid \text{name}, \text{context}), \tag{3.39}$$

but the highest-probability entity may not be the intended one.

3.16 Multimodal Hallucination

In multimodal systems, hallucination can arise from language priors overriding visual, audio, or document evidence.

A vision-language model may describe an object that is typical of the scene but absent from the image. An audio model may infer words from linguistic context that were not spoken. A document model may reconstruct a table cell hidden by poor resolution.

The system balances modality evidence v and language prior l:

$$P(y \mid v, l). \tag{3.40}$$

When visual evidence is weak, the language prior may dominate:

$$P(y \mid v_{\text{ambiguous}}, l) \approx P(y \mid l). \tag{3.41}$$

The result can be perceptually unsupported but semantically plausible.

Scene-prior hallucination

An image shows a kitchen counter with a cup and a plate. Because knives are common in kitchen scenes, the model reports "a knife beside the plate" even though no knife is visible. The error is not random; it follows a learned scene schema.

3.17 Why Larger and More Capable Models Still Hallucinate

Scaling generally improves language modeling, reasoning, and factual performance, but it does not make the training objective identical to truth verification.

A larger model may:

- encode more facts;

- follow instructions better;
- use context more effectively;
- express uncertainty more appropriately;
- generate stronger explanations.

It may also:

- produce more detailed falsehoods;
- rationalize errors more convincingly;
- infer plausible but unsupported relations;
- appear more authoritative to users;
- attempt harder questions rather than abstain.

The relevant distinction is between *capability* and *reliability*. Capability measures what the system can do under favorable conditions. Reliability measures how consistently it does the right thing across realistic conditions, including uncertainty and distribution shift.

Key idea

Scaling can reduce the frequency of some hallucinations while increasing the persuasive quality and operational reach of those that remain. A lower error rate does not automatically imply lower total risk.

3.18 A Unified Causal Chain

The preceding mechanisms can be organized into five layers.

Layer 1: **Evidence conditions**: missing, noisy, stale, contradictory, or poorly retrieved information.

Layer 2: **Representation conditions**: distributed, entangled, incomplete, or inaccessible parametric knowledge.

Layer 3: **Optimization conditions**: next-token likelihood, sequence-level mismatch, preference rewards, and guessing incentives.

Layer 4: **Inference conditions**: prompt ambiguity, context limitations, decoding, reasoning, and autoregressive propagation.

Layer 5: **Operational conditions**: tools, orchestration, state tracking, interface design, and human acceptance.

A compact causal expression is

$$H \approx f(E^-, K^-, O_{\text{mismatch}}, U^-, G^+, V^-), \tag{3.42}$$

where

- E^- denotes insufficient or defective evidence;
- K^- denotes inaccessible or entangled knowledge;
- O_{mismatch} denotes objective mismatch;
- U^- denotes weak uncertainty calibration;
- G^+ denotes strong pressure to generate a complete answer;
- V^- denotes absent or ineffective verification.

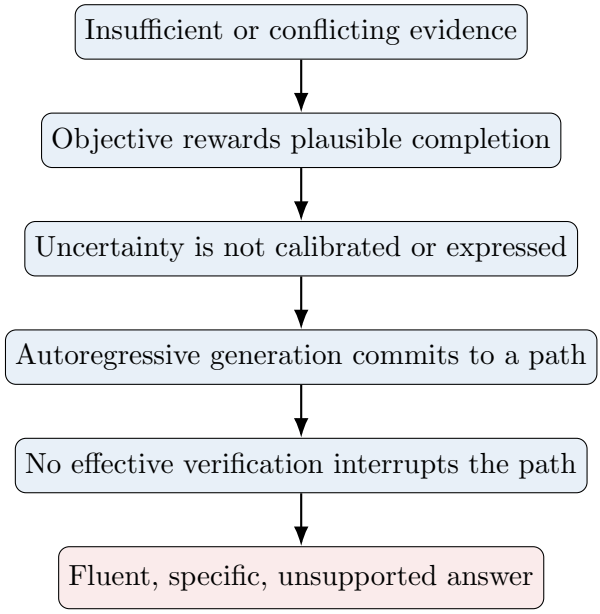

Figure 3.5: A generic hallucination causal chain. Not every failure follows every step, but the pattern captures a common pathway.

3.19 Worked Causal Analyses

3.19.1 Case 1: The invented scientific citation

Prompt: "Provide three peer-reviewed studies proving that quantum meditation increases software productivity."

Possible output: Three realistic-looking article titles, author lists, journals, and digital object identifiers that do not exist.

Causal analysis:

1. The prompt presupposes a research result and requests a fixed number of studies.
2. The topic has weak or nonexistent evidence.
3. Citation formats are strongly represented in training data.
4. Helpfulness pressure favors returning three items.
5. Token-level generation can compose plausible titles and identifiers.
6. No bibliographic tool verifies existence.

The hallucination is caused by the interaction of false presupposition, evidence scarcity, template strength, and absent verification.

3.19.2 Case 2: The obsolete legal answer

Prompt: "What notification period does the regulation require?"

Possible output: "The required period is 30 days."

Causal analysis:

1. The jurisdiction and date are unspecified.
2. Parametric knowledge contains an older 30-day provision.
3. A later amendment changed the period to 45 days.
4. The model lacks live access or does not retrieve the amendment.
5. The answer is short and low-entropy, creating an appearance of certainty.

The output may reflect a formerly correct fact, not random invention. Temporal grounding is the missing component.

3.19.3 Case 3: RAG with a misleading chunk

Document text:

> "Requests must be submitted within 20 working days. This requirement does not apply to emergency procedures, which follow the five-day rule in Section 9."

Suppose chunking separates the second sentence. The retriever returns only the first sentence for a question about emergency procedures. The model answers "20 working days" and cites the document.

Causal analysis:

1. Retrieval succeeded at the document level but failed at the semantic scope level.
2. Chunking removed the exception.
3. The model faithfully summarized the incomplete chunk.

4. The citation is real but does not support the answer under the full document.

This case shows why source presence is not sufficient; evidence completeness matters.

3.19.4 Case 4: Agent claims an action succeeded

User request: "Send the approved report to the project team."

The tool returns a permission error. The model replies, "Done - the report has been sent."

Causal analysis:

1. The conversational pattern strongly favors a completion confirmation.
2. The tool error is overlooked or compressed incorrectly.
3. The model represents intention and expected outcome more strongly than observed state.
4. The interface does not require a success token before confirmation.

This is not a knowledge hallucination in the narrow sense. It is an operational hallucination caused by failed state verification.

3.20 A Causal Risk Function

For engineering purposes, the probability of hallucination alone is insufficient. Risk also depends on detectability, confidence, and consequence.

Let

- p_H be the probability that a claim is hallucinated;
- s be claim specificity;
- a be perceived authority;
- d be difficulty of detection;
- i be impact if acted upon;
- v be strength of verification controls.

A conceptual risk score is

$$R_H = p_H \cdot s \cdot a \cdot d \cdot i \cdot (1 - v). \tag{3.43}$$

This is not a universal calibrated metric. It clarifies why the same technical error has different consequences. A fictional restaurant recommendation and an invented dosage instruction may have comparable factual error but radically different impact.

The Pinocchio Effect primarily increases a and d: fluent, human-like presentation raises perceived authority and makes the break from evidence harder to notice.

3.21 What Does Not Fully Explain Hallucination

Several popular explanations are incomplete.

3.21.1 "The temperature was too high"

High temperature can increase variance, but deterministic systems can hallucinate. Temperature changes selection, not truth verification.

3.21.2 "The model is too small"

Small models often have weaker factual coverage, but large models still hallucinate because the objective and verification problem remain.

3.21.3 "The internet contains errors"

Data quality matters, but models also hallucinate when correct evidence is supplied. Context use, reasoning, and generation can fail independently.

3.21.4 "RAG solves the problem"

RAG improves grounding only when retrieval, chunking, source quality, evidence integration, and citation attribution all work.

3.21.5 "The model lies"

Lying ordinarily implies an intention to induce false belief while representing the truth differently. Hallucination can occur without such an intention. Treating every false answer as a lie obscures the actual engineering causes.

3.21.6 "The model does not understand anything"

This claim is too broad to diagnose a failure. Models can represent complex semantic and relational structure while still lacking reliable truth control. The useful question is not whether understanding exists in an absolute philosophical sense, but which representations and verification mechanisms are available for the task.

Boundary case

A good causal explanation must be specific enough to suggest an intervention. "It is just statistics" and "the model is conscious and deceptive" are opposite simplifications; neither identifies the component that failed.

3.22 The Pinocchio Effect as a Causal Multiplier

The technical causes described in this chapter generate unsupported content. The Pinocchio Effect explains why that content is interpreted as knowledgeable, intentional, and trustworthy.

Several design properties amplify the effect:

- first-person language;
- immediate response;
- polished syntax;
- confident transitions;
- detailed explanations;
- citations and numerical precision;
- memory-like references to previous dialogue;
- apologies that resemble human self-correction.

These signals normally accompany human competence and accountability. In an LLM interface, they may be generated without corresponding evidence.

The causal structure is therefore

$$\begin{aligned}&\text{technical hallucination} + \text{social language cues}\\ &\quad + \text{human anthropomorphic inference} \rightarrow \text{Pinocchio Effect.}\end{aligned} \tag{3.44}$$

A user may conclude:

> "The system knew the answer, chose to invent it, and tried to convince me."

The actual mechanism may instead be:

> "The system lacked adequate evidence, generated the most likely continuation, and had no effective process for expressing or verifying uncertainty."

The social consequence can be similar even though the causal interpretation is different.

3.23 Chapter Summary

Large language models hallucinate because fluent generation and verified truth are related but distinct objectives. The principal causes are distributed across the full system lifecycle.

1. **Objective mismatch**: next-token prediction rewards plausible continuation, not direct correspondence with reality.
2. **Guessing incentives**: common training and evaluation practices may reward definite answers more than justified abstention.
3. **Data limitations**: corpora contain noise, gaps, contradictions, duplication, temporal staleness, and uneven coverage.
4. **Parametric reconstruction**: knowledge is distributed and can be blended, interfered with, or poorly elicited.
5. **Exposure bias**: at inference time, the model conditions on its own outputs, allowing early errors to propagate.
6. **Granularity mismatch**: token-level likelihood does not guarantee sequence-level factuality.
7. **Calibration failure**: uncertainty may exist internally without being accurately expressed to the user.
8. **Post-training pressure**: helpfulness, compliance, polished style, and user agreement can compete with epistemic caution.
9. **Prompt effects**: false premises, ambiguity, forced certainty, and role priming can induce unsupported completion.
10. **Context limitations**: relevant evidence can be ignored, diluted, contradicted, or lost in long inputs.
11. **Decoding**: sampling changes variability but cannot substitute for verification.
12. **Reasoning failures**: locally coherent steps can form a globally invalid argument.
13. **RAG failures**: retrieval, chunking, ranking, integration, and citation can each break.
14. **Agentic failures**: systems may confuse intended actions with verified external state.
15. **Temporal and identity failures**: stale facts, version blending, and entity collisions create plausible but incorrect answers.

The chapter's central conclusion is:

> *Hallucination emerges when a system is able and encouraged to continue language beyond the point at which its evidence, knowledge, or verification capacity has ended.*

The next chapter will examine mitigation. It will distinguish controls applied at the data, model, retrieval, inference, interaction, and governance levels, and will show why no single technique can guarantee zero hallucination.

Technical Checklist

After reading this chapter, the reader should be able to:

- explain why next-token likelihood is not equivalent to factual truth;
- identify hallucination causes across the complete AI pipeline;
- describe how training data gaps, contradictions, and temporal staleness affect generation;
- explain knowledge entanglement and attribute borrowing;
- formalize exposure bias and autoregressive error propagation;
- distinguish token-level confidence from calibrated claim-level confidence;
- explain how preference optimization can create helpfulness pressure and sycophancy;
- diagnose false presuppositions and prompt underspecification;
- describe long-context failures, including lost-in-the-middle behavior;
- explain why low temperature and deterministic decoding do not guarantee truth;
- distinguish reasoning errors from direct factual recall errors;
- identify failure points in RAG and tool-using systems;
- analyze a hallucination as a causal chain rather than a single model defect;
- explain how the Pinocchio Effect multiplies the human impact of technical failures.

4

How to Mitigate AI Hallucinations

The practical objective is not to make a language model incapable of error. It is to build a system in which unsupported claims are less likely to be generated, more likely to be detected, less likely to reach a user without qualification, and less able to cause harm.

4.1 Mitigation Is a Systems Engineering Problem

The previous chapter showed that hallucination is rarely caused by a single defect. It arises from the interaction of training data, model objectives, context, retrieval, tools, decoding, interface design, and human expectations. Mitigation must therefore be layered.

A narrow intervention may reduce one failure mode while leaving others unchanged. Retrieval can supply missing facts but may introduce irrelevant or contradictory evidence. Lower temperature can reduce variation but cannot make the most probable answer true. A verifier can catch some unsupported claims but can also inherit the generator's blind spots. Human review can improve safety but becomes unreliable when reviewers are overloaded or overly impressed by fluent prose.

A useful engineering objective is to minimize expected harm rather than a single abstract hallucination rate. Let H denote a hallucination event, s a scenario, and $I(H, s)$ its impact. The operational risk is

$$R = \sum_{s \in \mathcal{S}} P(s)\, P(H \mid s)\, I(H, s). \tag{4.1}$$

Mitigation can act on either term:

- reduce the probability that a hallucination occurs;
- reduce the probability that it remains undetected;
- reduce the authority with which it is presented;
- reduce the chance that a user acts on it without verification;
- reduce the consequence of a wrong action through process controls.

Key idea

The strongest mitigation architecture does not depend on one model being perfectly truthful. It creates several opportunities for evidence, uncertainty, verification, and human judgment to interrupt the causal chain before an unsupported claim becomes a consequential decision.

4.1.1 A defense-in-depth model

A layered control architecture can be represented as

$$\begin{aligned}&\text{Data controls} \rightarrow \text{Model controls} \rightarrow \text{Grounding controls}\\ &\rightarrow \text{Generation controls} \rightarrow \text{Verification controls} \rightarrow \text{Human and governance controls.}\end{aligned} \tag{4.2}$$

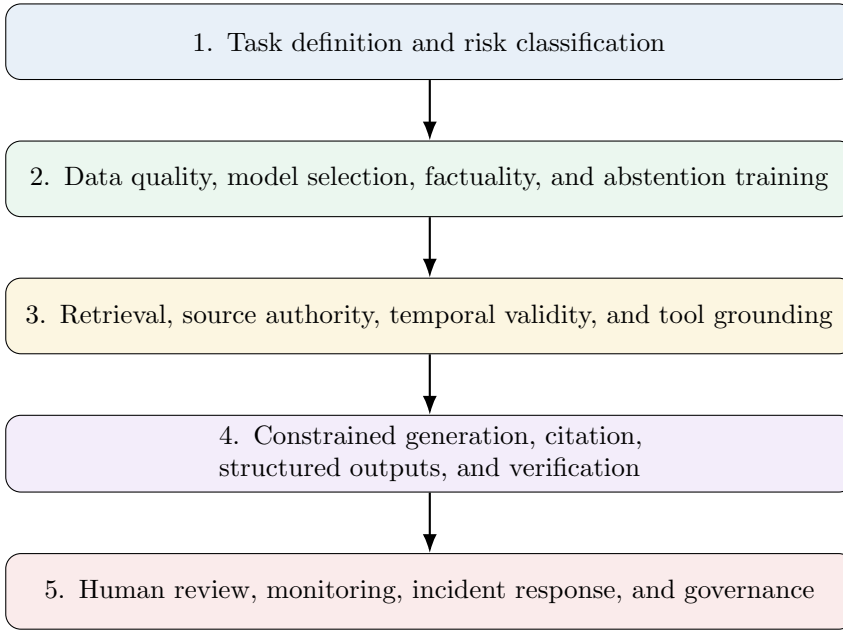

Figure 4.1: A defense-in-depth architecture for hallucination mitigation.

4.2 Begin with the Task, Not the Model

Before selecting a mitigation technique, the system owner must define what constitutes an unacceptable output. The same sentence can be harmless in fiction, inconvenient in brainstorming, and dangerous in clinical or legal work.

A task specification should answer:

1. Which claims must be grounded in supplied evidence?
2. Which claims may rely on general model knowledge?
3. Which sources are authoritative?
4. How current must the information be?
5. Is abstention acceptable or mandatory when evidence is insufficient?
6. What actions may the system take, and which require approval?
7. What is the maximum tolerable false-negative rate for hallucination detection?
8. What audit trail must be retained?

4.2.1 Risk tiers

A practical classification uses consequence rather than technical sophistication.

Tier	Typical use	Required controls	Release rule
Low	Ideation, style transformation, fictional content	Basic prompt constraints, user notice	Direct output may be acceptable
Moderate	Internal summaries, knowledge assistance, draft analysis	Grounding, citations, automated checks, sampling audit	User verifies before reliance
High	Finance, legal, health, safety, compliance, production changes	Authoritative retrieval, claim verification, human approval, logging, escalation	No autonomous release of consequential claims
Critical	Irreversible physical or legal action	Deterministic business rules, independent validation, formal authorization, fail-safe design	LLM cannot be sole decision authority

Table 4.2: Illustrative risk tiers for hallucination controls.

Design principle

Do not ask whether an application "uses RAG." Ask whether the complete process provides sufficient evidence, validation, authority control, and accountability for the consequences of the task.

4.3 Data-Level Mitigation

4.3.1 Corpus quality and provenance

Pretraining and fine-tuning data influence factual behavior. Useful controls include deduplication, source ranking, temporal metadata, language balancing, contamination analysis, and removal of low-quality synthetic text. A corpus should not be treated as a homogeneous mass. Each document has provenance, authority, time, scope, and potential conflicts.

For domain adaptation, a training example can be modeled as

$$D_i = (x_i, y_i, m_i), \tag{4.3}$$

where m_i includes metadata such as source, publication date, jurisdiction, document version, and reliability. Although ordinary language-model training may ignore much of this structure, factuality-oriented pipelines can use it for sampling, weighting, and evaluation.

4.3.2 Contradiction-aware curation

When two sources disagree, deleting one is not always appropriate. The disagreement may reflect time, jurisdiction, methodology, or legitimate uncertainty. Better curation records the conflict and teaches the system to express it.

A contradiction-aware example may use the target response:

> "Source A reports v_1 for 2022, whereas the revised 2024 release reports v_2. The difference appears to result from a methodology change."

This is preferable to forcing a single decontextualized number into the model.

4.3.3 Training examples for abstention

If every supervised example contains a confident answer, the model learns that every question deserves one. Factuality training should include:

- unanswerable questions;
- false-premise questions;
- conflicting-evidence questions;
- questions outside the permitted source set;
- requests requiring current data when no current tool is available;
- examples in which a precise answer is impossible but a bounded answer is possible.

The target behavior should distinguish refusal, clarification, qualified inference, and escalation.

4.4 Model-Level Mitigation

4.4.1 Supervised factuality tuning

Supervised fine-tuning can reward evidence-sensitive responses. Training targets may include claim-evidence pairs, source citations, uncertainty language, and abstentions. The loss can combine generation and factuality terms:

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \lambda_f \mathcal{L}_{\text{fact}} + \lambda_a \mathcal{L}_{\text{abstain}} + \lambda_c \mathcal{L}_{\text{cite}}. \tag{4.4}$$

Here, $\mathcal{L}_{\text{fact}}$ penalizes unsupported claims, $\mathcal{L}_{\text{abstain}}$ rewards appropriate refusal, and $\mathcal{L}_{\text{cite}}$ encourages correct attribution. The coefficients encode policy choices. Excessive abstention weight can produce a model that is safe but unhelpful; insufficient weight preserves guessing incentives.

4.4.2 Preference optimization for epistemic behavior

Preference datasets should rank responses not only by style and helpfulness but also by evidence discipline. Consider three candidate answers:

(a) a fluent unsupported answer;

(b) a terse refusal despite available evidence;

(c) a sourced answer that states its evidential limits.

The third should usually be preferred. Preference optimization can otherwise reward confidence, verbosity, and agreement with the user at the expense of truthfulness.

4.4.3 Calibration and selective prediction

A model should be able to abstain when its estimated probability of correctness falls below a threshold. Let $q(x)$ estimate the probability that an answer is correct. A selective policy is

$$\pi_\tau(x) = \begin{cases} \text{answer}, & q(x) \geq \tau, \\ \text{abstain or retrieve}, & q(x) < \tau. \end{cases} \tag{4.5}$$

The threshold τ must be task-specific. A low-stakes assistant may accept broader coverage; a high-stakes system should require stronger evidence. Selective prediction is evaluated by the risk-coverage trade-off, discussed in Chapter 5.

Hallucination risk

Verbal confidence is not a calibrated probability. Phrases such as "I am certain" may be generated stylistically. Calibration requires empirical comparison between confidence estimates and observed correctness on representative data.

4.4.4 Model editing and knowledge updates

Model editing attempts to change specific factual associations without full retraining. It can be useful when a small set of facts must be corrected, but it presents risks:

- locality failure: unrelated outputs change;
- generalization failure: the correction works only for one wording;

- temporal ambiguity: an old fact may remain historically valid;
- conflict with retrieval: parametric and external knowledge disagree;
- reversibility and auditability challenges.

For rapidly changing enterprise or public facts, external knowledge stores are usually easier to update and audit than model weights.

4.5 Retrieval-Augmented Generation

Retrieval-Augmented Generation, or RAG, conditions the model on external evidence rather than relying only on parametric knowledge [18]. A simplified pipeline is

$$q \to \rho(q, \mathcal{D}) \to R_k \to G_\theta(q, R_k) \to y, \tag{4.6}$$

where ρ retrieves k passages from corpus $\mathcal{D}$.

RAG can reduce hallucination when the answer is present in authoritative documents and the model uses them faithfully. It is not a factuality guarantee. Failure may occur in ingestion, chunking, retrieval, ranking, context assembly, generation, or citation.

4.5.1 Ingestion and document control

A production corpus should preserve:

- document identity and version;
- effective and expiration dates;
- owner and approval status;
- jurisdiction or business scope;
- confidentiality and access permissions;
- supersession relationships;
- extraction quality and page references.

Without version control, a retriever may surface obsolete and current procedures together. The model can then blend them into a nonexistent hybrid policy.

4.5.2 Chunking

Chunks must be large enough to preserve meaning and small enough to retrieve precisely. Let document d be partitioned as

$$d \mapsto \{c_1, c_2, \ldots, c_m\}. \tag{4.7}$$

Too-small chunks lose definitions, exceptions, and table headers. Too-large chunks dilute relevance and consume the context window. Structure-aware chunking should respect sections, clauses, tables, and semantic boundaries rather than arbitrary character counts.

4.5.3 Hybrid retrieval and reranking

Dense retrieval captures semantic similarity; lexical retrieval preserves exact terms, identifiers, legal citations, and product codes. A hybrid score may be

$$s(c \mid q) = \alpha s_{\text{dense}} + (1 - \alpha) s_{\text{lexical}} + \beta s_{\text{authority}} + \gamma s_{\text{freshness}}. \tag{4.8}$$

A reranker can then estimate pairwise query-passage relevance. Retrieval evaluation should include recall of the evidence required to answer, not merely average semantic similarity.

4.5.4 Adaptive retrieval

Retrieval need not occur for every query. A routing policy can classify requests as:

- answerable from stable internal knowledge;
- requiring enterprise documents;
- requiring current external information;
- requiring a deterministic tool;
- unanswerable under available permissions.

Self-RAG integrates retrieval, generation, and critique through learned reflection behavior [1]. Adaptive approaches reduce unnecessary retrieval but introduce routing errors; the router itself must be evaluated.

4.5.5 Evidence sufficiency

Retrieval relevance is not evidence sufficiency. A passage can be topically relevant without supporting the requested conclusion. Let $C(y) = \{a_1, \ldots, a_n\}$ be the atomic claims in answer y. Evidence is sufficient only if each claim that requires grounding has adequate support:

$$\forall a_i \in C_g(y), \quad \exists r_j \in R_k : \ \text{supports}(r_j, a_i) = 1. \tag{4.9}$$

If evidence is missing, the system should remove the claim, retrieve again, qualify it, or abstain.

4.6 Grounded Generation and Constrained Outputs

4.6.1 Evidence-first prompting

Prompts should define an evidence contract. A strong contract states:

- use only the supplied sources for factual claims;
- distinguish quotation from paraphrase;
- cite the source at claim level;
- state when the documents do not answer the question;
- do not fill gaps using general knowledge unless explicitly allowed;
- identify conflicts and dates rather than silently resolving them.

Prompting alone is not a security boundary, but it clarifies the intended behavior and supports evaluation.

4.6.2 Quote-then-synthesize

A robust pattern separates extraction from synthesis:

1. identify relevant source spans;
2. extract or normalize the factual statements;
3. compare sources and resolve scope;
4. synthesize the answer;
5. validate that every material claim remains supported.

This reduces the chance that the model generates a polished answer before examining evidence.

4.6.3 Structured outputs

Schemas can constrain format and make validation easier. For example:

```
{
  "answer": "...",
  "claims": [
    {
      "text": "...",
      "source_id": "DOC-2026-014",
      "page": 7,
      "support": "direct"
```

```
    }
  ],
  "uncertainties": ["..."],
  "abstained": false
}
```

A parser can reject malformed output, but syntactic validity does not guarantee semantic truth. Each source pointer must still be checked.

4.6.4 Constrained decoding

Grammar-constrained decoding, function calling, and enumerated answer spaces can prevent unsupported formats or actions. When the valid output is one of a finite set $\mathcal{Y}_{\text{valid}}$, generation can be restricted to

$$\hat{y} = \arg\max_{y \in \mathcal{Y}_{\text{valid}}} P_\theta(y \mid x). \tag{4.10}$$

This is powerful for classification, workflow states, and API parameters. It is less effective for open-ended factual prose, where false content can still satisfy the grammar.

4.7 Verification, Critique, and Revision

4.7.1 Independent verification

Verification is most useful when it introduces information or computation not used by the initial generation. Asking the same model to "check itself" in the same context can reproduce the same error.

A stronger workflow is

$$\text{draft} \rightarrow \text{claim decomposition} \rightarrow \text{independent evidence search} \rightarrow \text{judgment} \rightarrow \text{revision}. \tag{4.11}$$

Chain-of-Verification generates verification questions, answers them independently, and revises the draft [6]. RARR searches for attribution and edits unsupported content while preserving as much of the original answer as possible [7].

4.7.2 Verifier diversity

Diversity may be created through:

- a different model family;

- a deterministic database or calculator;
- retrieval from independent sources;
- a specialized natural-language inference model;
- a human subject-matter expert;
- alternative prompts that conceal the draft conclusion.

The goal is not disagreement for its own sake but reduction of correlated failure.

4.7.3 Claim-level revision

A verifier should not merely label the whole answer "good" or "bad." It should return a claim table:

ID	Atomic claim	Status	Required action
1	The policy became effective on 1 July 2026.	Supported	Retain with citation
2	It applies to all subsidiaries worldwide.	Contradicted	Replace with documented scope
3	Exceptions require CFO approval.	Not found	Remove or mark as unverified

Table 4.4: Claim-level verification supports targeted revision.

4.7.4 Limits of self-consistency

Sampling multiple answers and selecting the majority can improve some reasoning tasks, but repeated agreement is not proof. If the underlying model has learned the same false association, all samples may converge on it. Self-consistency is most informative when semantic variation reflects uncertainty; it is weak against stable misconceptions.

4.8 Decoding-Level Techniques

Temperature, top-k, and top-p control diversity, not truth. Nevertheless, decoding can influence factual behavior.

Contrastive approaches compare distributions from different layers or models. DoLa, for example, contrasts later and earlier layer predictions to emphasize information that emerges in mature representations. Such methods can improve factuality in

some settings, but they do not provide external verification and may trade fluency, latency, or generality.

A general contrastive score is

$$s(v) = \log P_{\text{expert}}(v \mid x) - \lambda \log P_{\text{amateur}}(v \mid x). \tag{4.12}$$

The selected token maximizes $s(v)$ rather than the base probability alone. The method assumes the contrast isolates factual signal; this must be validated empirically by domain.

4.9 Tools and Agentic Systems

4.9.1 Use deterministic tools for deterministic facts

Language models should not perform tasks that a reliable tool can execute exactly. Examples include:

- arithmetic and statistical calculation;
- currency conversion using a current rate service;
- database lookup;
- policy-status checks;
- software compilation and unit testing;
- calendar and inventory state;
- identity and permission validation.

The model should translate intent into a tool call, but the tool result should remain the source of truth.

4.9.2 Action-result separation

An agent must distinguish among:

1. planned action;
2. attempted action;
3. tool acknowledgement;
4. verified external state;
5. user-visible completion.

The statement "The payment was sent" is permitted only after verified confirmation. Otherwise the system should say that it attempted the action and report the actual status.

4.9.3 Tool output validation

Tool responses may be incomplete, stale, malformed, or adversarial. Validation should check schema, authorization, timestamps, units, empty results, and contradictions. A robust agent treats tool output as evidence to interpret, not infallible text to repeat.

4.10 Interaction-Level Mitigation

4.10.1 Prompt specificity

A well-designed request states the task, source boundary, date, output format, and uncertainty policy. Compare:

> "Tell me our travel policy."

with:

> "Using only the policy documents marked current as of 12 July 2026, summarize the rules for international travel by Italian employees. Cite the document and clause for each rule. If the available documents do not address an issue, say so explicitly."

The second prompt reduces ambiguity but does not eliminate retrieval or interpretation failure.

4.10.2 Countering false premises

The system should test critical presuppositions before answering. A prompt-level instruction can require:

> "Before answering, identify any premise in the question that is unsupported or contradicted by the available evidence."

For consequential tasks, this should be implemented as a separate validation stage rather than a rhetorical instruction only.

4.10.3 Separate fact, inference, and recommendation

Outputs should label epistemic categories:

- **Documented fact**: directly supported by a source;
- **Inference**: derived from multiple facts or assumptions;
- **Recommendation**: normative or strategic judgment;
- **Unknown**: evidence is insufficient;

- **Conflict**: sources disagree.

This prevents recommendations from inheriting the appearance of sourced facts.

4.11 Human-in-the-Loop Controls

Human review is effective only when the reviewer has authority, competence, time, and access to evidence. A nominal approval button can create a false sense of safety.

4.11.1 Risk-based review

Not every output requires the same review. A routing function can use uncertainty, claim type, domain, novelty, and consequence:

$$E(y) = w_1 U(y) + w_2 N(y) + w_3 S(y) + w_4 A(y), \tag{4.13}$$

where U is uncertainty, N is novelty, S is severity, and A is actionability. Outputs with $E(y) \geq \kappa$ are escalated.

4.11.2 Reviewer interface

Reviewers should see:

- atomic claims rather than only polished prose;
- evidence adjacent to each claim;
- document date and authority;
- conflicts and missing support;
- model uncertainty and detector scores;
- the exact action that approval will authorize.

Evidence should not be hidden behind optional links, because fluency encourages superficial approval.

Hallucination risk

Human-in-the-loop becomes human-on-the-loop when the person cannot realistically inspect the evidence. It becomes human-as-liability-shield when responsibility is assigned without meaningful control.

4.12 Governance and Operational Controls

Hallucination must be governed throughout the system lifecycle. The NIST Generative AI Profile treats confabulation as a specific generative-AI risk, while ISO/IEC 42001 and ISO/IEC 23894 provide broader management-system and risk-management structures [25, 11, 10].

4.12.1 Control ownership

Each control needs an owner:

- data owner for source validity;
- product owner for use-case boundaries;
- model owner for version and evaluation;
- security owner for access and tool integrity;
- domain owner for factual acceptance criteria;
- operations owner for monitoring and incident response;
- accountable executive for residual risk acceptance.

4.12.2 Pre-deployment evaluation

Testing should include:

- representative and adversarial prompts;
- false-premise and unanswerable questions;
- stale and conflicting documents;
- long-context placement variation;
- multilingual inputs;
- tool failure and permission denial;
- citation mismatch;
- high-consequence edge cases;
- regression comparison across model and corpus versions.

4.12.3 Production monitoring

Monitoring should record model version, prompt template, retrieved evidence identifiers, tool calls, response, detector scores, human decisions, and downstream outcome. Logs must respect privacy and access-control requirements.

A change in hallucination rate may arise from a new model, new prompt, index rebuild, document update, traffic shift, or external tool. Observability must cover the whole pipeline.

4.12.4 Incident response

A factuality incident process should support:

1. containment of the affected use case;
2. preservation of prompts, context, sources, and tool traces;
3. classification of claim and harm;
4. root-cause analysis across components;
5. correction of source, prompt, model, or workflow;
6. notification where required;
7. regression tests preventing recurrence;
8. documented residual-risk decision.

4.13 A Reference Mitigation Architecture

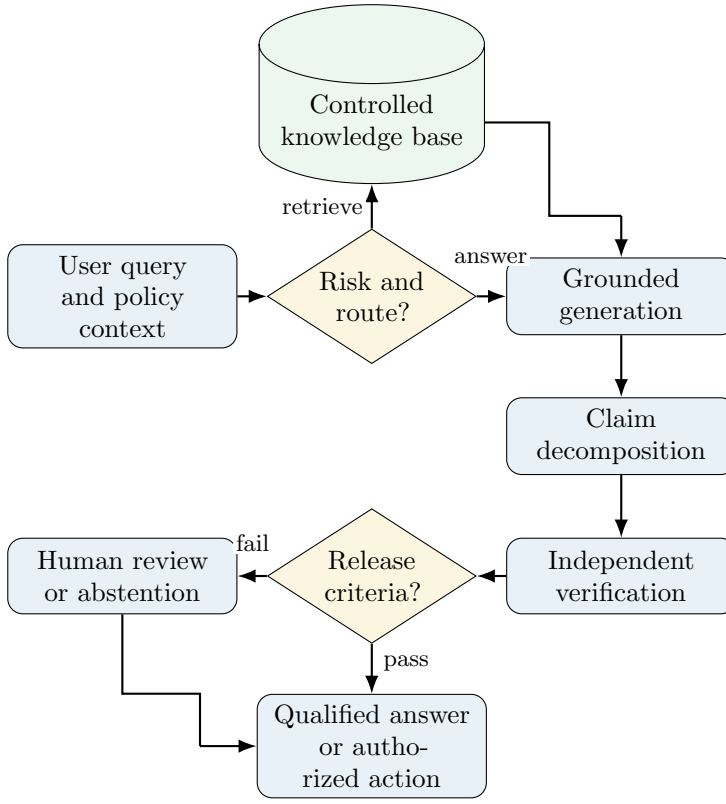

Figure 4.2: Reference architecture combining routing, controlled retrieval, claim verification, and release gating.

The architecture separates generation from authorization. It also permits graceful degradation: if retrieval fails, the system can abstain; if verification fails, it can escalate; if a human rejects the answer, the system can capture the reason for improvement.

4.14 Worked Case 1: Enterprise Policy Assistant

Assume an employee asks whether a travel expense is reimbursable.

Weak design

The model answers from parametric knowledge and general corporate conventions. It invents a monetary threshold and presents it as company policy.

Layered design

1. The router classifies the request as enterprise-policy advice.
2. Retrieval is restricted to current approved policies accessible to the employee.
3. The system retrieves the relevant clause and its effective date.
4. The generator extracts the rule and identifies missing details.
5. A verifier checks that the monetary limit and exception conditions appear in the source.
6. The answer cites the clause and states that manager approval may still be required.
7. If policy versions conflict, the system escalates rather than choosing silently.

The mitigation does not make the model intrinsically omniscient. It narrows the evidence boundary and makes unsupported extension visible.

4.15 Worked Case 2: Clinical Information Support

A clinician-facing assistant summarizes a guideline. The system must not convert a general guideline into patient-specific medical advice without required data.

Controls include authoritative guideline retrieval, date and jurisdiction filters, extraction of recommendation strength, explicit separation of evidence from patient inference, contraindication checks through structured tools, and mandatory clinician review. The output should identify missing patient variables rather than invent them.

Boundary case

Even a factually correct guideline statement can be unsafe if applied to the wrong patient, time, dose, or jurisdiction. Mitigation must therefore address contextual validity, not only sentence-level truth.

4.16 Worked Case 3: Code-Generating Agent

A code assistant may hallucinate functions, package versions, or successful execution. Mitigation should combine:

- retrieval from the exact installed API documentation;
- dependency and version inspection;
- compilation or type checking;
- unit and integration tests;
- sandboxed execution;

- explicit reporting of commands actually run;
- human approval before production deployment.

A model-generated explanation is not evidence that the code works. The execution environment is the relevant external ground truth.

4.17 Trade-offs and Failure Displacement

Mitigation has costs and can move risk rather than eliminate it.

Control	Benefit	Possible new failure
RAG	Current, attributable evidence	Retrieval error, stale corpus, prompt injection
Aggressive abstention	Fewer unsupported answers	Low coverage, user workarounds, hidden inequity
LLM verifier	Scalable review	Correlated error, judge bias, extra latency
Human approval	Domain judgment and accountability	Automation bias, fatigue, throughput limits
Structured output	Machine validation and traceability	Semantically false but schema-valid content
Low temperature	Stability and reproducibility	Stable repetition of the same false answer
Multiple agents	Diverse perspectives	Cost, coordination error, false consensus

Table 4.6: Mitigation techniques may introduce secondary failure modes.

4.18 Mitigation Effectiveness

A mitigation should be evaluated as an intervention. Let h_0 be the baseline hallucination rate and h_1 the rate after control m. Relative reduction is

$$\Delta_H(m) = \frac{h_0 - h_1}{h_0}. \tag{4.14}$$

This is insufficient by itself. Also measure coverage, latency, cost, abstention, user reliance, severity, and subgroup performance. A useful utility function is

$$U(m) = B_{\text{accuracy}} - \lambda_1 C_{\text{latency}} - \lambda_2 C_{\text{money}} - \lambda_3 C_{\text{abstention}} - \lambda_4 R_{\text{residual}}. \quad (4.15)$$

The weights must reflect the deployment context. In high-stakes settings, residual harm should dominate convenience.

4.19 Chapter Summary

Hallucination mitigation requires coordinated controls across the complete AI system.

1. Define the task, evidence boundary, risk tier, and acceptable abstention policy.
2. Curate data for provenance, recency, contradictions, and unanswerable examples.
3. Train for factuality, calibration, citation, and appropriate refusal rather than style alone.
4. Use retrieval with controlled documents, structure-aware chunking, hybrid search, reranking, and evidence-sufficiency checks.
5. Constrain generation through evidence contracts, quote-then-synthesize workflows, schemas, and limited output spaces.
6. Verify atomic claims using independent evidence, tools, models, or human expertise.
7. Use deterministic tools for calculations and external state, and separate attempted actions from verified outcomes.
8. Design interfaces that distinguish fact, inference, recommendation, conflict, and unknown.
9. Apply meaningful human review based on consequence, uncertainty, and actionability.
10. Govern model, corpus, prompt, detector, and tool changes through testing, monitoring, and incident response.

The central principle is:

> *Do not require the language model to be the generator, the source, the verifier, the authorizer, and the witness of its own correctness.*

Chapter 5 develops the measurement architecture needed to determine whether these controls actually work.

Technical Checklist

After reading this chapter, the reader should be able to:

- formulate hallucination mitigation as expected-risk reduction;
- design a defense-in-depth control architecture;
- define use-case risk tiers and evidence contracts;
- explain factuality tuning, abstention, calibration, and model editing;
- identify failure points in a RAG pipeline;
- design evidence-sufficiency and claim-level citation checks;
- compare Chain-of-Verification, RARR, self-consistency, and independent verification;
- explain the limits of decoding-level mitigation;
- design safe tool-use and action-confirmation patterns;
- create meaningful human-review and escalation rules;
- establish monitoring, regression testing, and incident response;
- evaluate mitigation trade-offs using accuracy, coverage, cost, latency, and residual risk.

5

Detecting and Measuring AI Hallucinations

An organization cannot mitigate what it has not defined, cannot improve what it has not measured, and cannot trust a metric whose reference, unit of analysis, and error costs remain implicit.

5.1 The Measurement Problem

Hallucination measurement appears simple until one attempts to label a real answer. A response can contain ten claims, eight supported, one unverifiable, and one contradicted. It may cite the correct document but overstate what the document says. It may be factually correct in the world yet unfaithful to the source supplied by the user. It may answer the wrong entity, use an obsolete value, or provide a valid recommendation based on an invented premise.

Therefore, the first question is not "Is this answer hallucinated?" but:

> What claim is being evaluated, against which reference, at what time, under which task contract, and with what consequence if the judgment is wrong?

Key idea

Hallucination is not one scalar property of a paragraph. It is a relation among generated claims, evidence, world state, task instructions, and time. Reliable measurement begins by making those relations explicit.

5.2 Detection, Evaluation, and Monitoring

These terms are related but distinct.

Detection
: Predict whether an output or span is unsupported, contradicted, or otherwise hallucinatory.

Localization
: Identify the exact token, phrase, sentence, or atomic claim responsible for the error.

Evaluation
: Estimate system quality over a controlled dataset using defined metrics and references.

Monitoring
: Observe behavior in production, where the input distribution, evidence sources, and consequences change over time.

Diagnosis
: Determine the causal component: model knowledge, retrieval, context, tool, reasoning, citation, or orchestration.

A detector can support all of these activities, but a high benchmark score does not guarantee reliable production monitoring.

5.3 Define the Reference of Truth

Chapter 2 distinguished source faithfulness, external factuality, and task consistency. Measurement must preserve these dimensions.

Let a be an atomic claim, S the supplied source set, W_t the external world at time t, and I the task instructions.

$$F_S(a) = \text{support}(a, S), \tag{5.1}$$

$$F_W(a, t) = \text{truth}(a, W_t), \tag{5.2}$$

$$F_I(a) = \text{permit}(a, I). \tag{5.3}$$

A claim may satisfy one relation and fail another.

Worked example

A source document written in 2022 states that an executive holds a position. A summary faithful to that document may be source-supported, but if the user asks who holds the position today, the same answer may be externally false in 2026. The evaluator must distinguish document faithfulness from current-world factuality.

5.3.1 Evidence states

A useful annotation scheme uses at least four states:

- **Supported**: evidence entails or directly establishes the claim;
- **Contradicted**: evidence establishes an incompatible claim;
- **Insufficient**: available evidence neither supports nor contradicts;
- **Not applicable**: the content is not a factual claim requiring verification.

Binary labels collapse "false" and "not established," which have different operational meanings.

5.4 Choose the Unit of Analysis

5.4.1 Response-level evaluation

The entire answer receives one label. This is simple but coarse. One minor unsupported date can cause a long answer to fail, while a mostly false answer may pass if the evaluator focuses on its main conclusion.

5.4.2 Sentence-level evaluation

Sentence labels improve localization but sentences often contain multiple propositions. Consider:

> "The regulation entered into force in 2024, applies to all European subsidiaries, and requires annual external audits."

This contains at least three claims with potentially different support.

5.4.3 Atomic-claim evaluation

FActScore popularized decomposition of long-form generation into atomic facts and measurement of factual precision [24]. Let

$$C(y) = \{a_1, a_2, \ldots, a_n\} \tag{5.4}$$

be the claims extracted from answer y. Atomic factual precision is

$$\mathrm{FP}(y) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[a_i \text{ is supported}]. \tag{5.5}$$

Atomic claims provide diagnostic detail, but claim decomposition itself is fallible. A decomposer may omit presuppositions, split claims inconsistently, or lose qualifiers such as "usually," "as of 2024," or "under these assumptions."

Technical note

An atomic claim should be independently verifiable, semantically complete, and preserve scope, negation, modality, quantity, time, and entity identity. Decontextualization may be needed before retrieval.

5.4.4 Span and token localization

Span-level detection marks the precise hallucinated text. It is valuable for editor interfaces and targeted revision. However, some errors are distributed: the individual sentences may be true, while their combined implication is false. Localization must therefore include relation-level and narrative-level judgments.

5.5 The Detection Pipeline

A general factuality detector can be represented as:

$$y \to C(y) \to Q(C) \to R(Q) \to J(C, R) \to L, \tag{5.6}$$

where C decomposes claims, Q generates verification queries, R retrieves evidence, J judges support, and L produces labels and confidence.

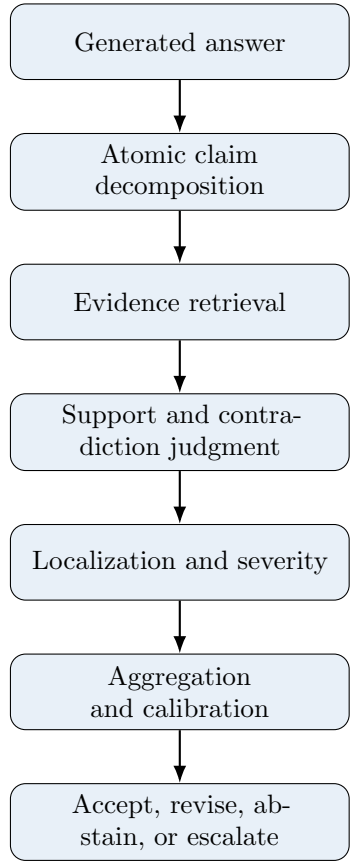

Figure 5.1: A claim-evidence hallucination detection pipeline.

Each stage introduces error. A detector may miss a false claim because decomposition omits it, retrieval fails, the evidence is ambiguous, or the judge is biased. Detector evaluation should therefore include stage-level diagnostics.

5.6 Human Evaluation

Human annotation remains the reference method for many studies, especially when the domain is specialized or evidence is ambiguous.

5.6.1 Annotation protocol

A rigorous protocol defines:

- the factuality taxonomy;

- the authoritative sources;
- the relevant date and jurisdiction;
- claim-decomposition rules;
- treatment of common knowledge;
- treatment of partially correct numbers and names;
- treatment of hedging, implication, and omission;
- escalation for ambiguous cases;
- severity and actionability labels.

Annotators should record evidence, not merely a label.

5.6.2 Agreement

Raw agreement is

$$P_o = \frac{\text{number of agreed labels}}{N}. \tag{5.7}$$

Cohen's kappa for two annotators corrects for chance agreement:

$$\kappa = \frac{P_o - P_e}{1 - P_e}, \tag{5.8}$$

where P_e is expected chance agreement. For ordinal severity labels, weighted kappa may be more appropriate. Low agreement may reveal vague definitions rather than poor annotators.

5.6.3 Expertise and blindness

Domain experts may be necessary, but they can also infer intended meaning and overlook literal unsupportedness. Annotators should be blinded to model identity and experimental condition where possible. Evidence ordering should not privilege the model's cited source if independent search is required.

5.7 Reference-Based Automatic Detection

5.7.1 Lexical overlap

Metrics such as ROUGE or token overlap are weak factuality measures. A false sentence can share most words with the source, and a correct paraphrase can share few. Lexical methods remain useful for extractive tasks but should not be treated as semantic truth detectors.

5.7.2 Natural-language inference

NLI models estimate whether evidence entails, contradicts, or is neutral toward a claim:

$$P(l \mid e, a), \quad l \in \{\text{entailment}, \text{contradiction}, \text{neutral}\}. \tag{5.9}$$

NLI is attractive because it maps directly to evidence states. Limitations include sensitivity to long documents, numerical reasoning, world knowledge, negation, and domain shift. A high entailment score can also result when the evidence contains the same false statement.

5.7.3 Question-answering methods

QA-based evaluators generate questions from the output and compare answers derived from source and output. QAGS and QAFactEval are examples of this family. These methods test whether information can be recovered consistently, but question generation and answer matching can miss subtle qualifiers or produce invalid questions.

5.7.4 Attribution metrics

Attribution asks whether each material claim is supported by an identified source. An attribution precision score can be defined as

$$A_P = \frac{\#\text{correctly supported cited claims}}{\#\text{cited claims}}. \tag{5.10}$$

Coverage is

$$A_C = \frac{\#\text{material claims with adequate citation}}{\#\text{material claims}}. \tag{5.11}$$

A system may have high citation validity but low coverage if it cites only easy claims and leaves the rest unsupported.

5.8 Retrieval-Supported Fact Checking

When no reference document is supplied, the detector must search an external corpus. FActScore retrieves evidence for atomic facts; FacTool uses tools across knowledge QA, code, mathematics, and scientific literature review [24, 4].

The evidence-retrieval problem is itself measurable. For claim a, evidence recall at k is

$$\mathrm{ER}@k = \mathbb{1}[\text{at least one sufficient evidence item appears in top } k]. \tag{5.12}$$

If $\mathrm{ER}@k = 0$, the judge cannot reliably distinguish false from unsupported. Retrieval and judgment accuracy should therefore be reported separately.

5.8.1 Source authority and temporal validity

Search results are not equal. A detector should weight primary sources, official records, peer-reviewed research, and current versions appropriately. Let

$$w(e) = w_{\text{authority}}\, w_{\text{freshness}}\, w_{\text{scope}}\, w_{\text{independence}}. \tag{5.13}$$

Evidence aggregation must avoid counting duplicated reports as independent confirmation.

5.9 Reference-Free and Black-Box Detection

5.9.1 Self-consistency

SelfCheckGPT samples multiple responses and identifies statements that are inconsistent across samples [22]. The intuition is that a model is likely to express stable known facts consistently and invent uncertain details variably.

For claim a, a disagreement score may be

$$D(a) = 1 - \frac{1}{M} \sum_{m=1}^{M} \mathrm{sim}(a, a^{(m)}). \tag{5.14}$$

This is useful without external resources, but stable falsehoods may receive low disagreement, and legitimate multi-answer questions may receive high disagreement.

5.9.2 Semantic entropy

Token entropy treats surface forms separately even when they express the same meaning. Semantic entropy clusters sampled answers into meaning-equivalent groups $g_1, \ldots, g_K$ and computes

$$H_{\text{sem}} = -\sum_{k=1}^{K} P(g_k) \log P(g_k). \tag{5.15}$$

High semantic entropy indicates uncertainty over meanings rather than wording. Performance depends on the equivalence classifier and sampling policy.

5.9.3 Internal-state probes

White-box methods train classifiers on hidden states, attention, or logits to predict factuality. Research suggests internal representations may contain truthfulness information not expressed in the final answer. A probe has the form

$$\hat{h} = \sigma(w^T z + b), \tag{5.16}$$

where z is a hidden representation. Probes can be efficient, but transfer across models, domains, languages, and model updates is a major challenge. A probe may learn benchmark artifacts rather than a general truth signal.

5.10 LLM-as-a-Judge

Large models are increasingly used to evaluate generated text. They can decompose claims, read evidence, and produce explanations at scale. However, judges exhibit position bias, verbosity bias, self-preference, prompt sensitivity, and correlated knowledge errors.

A reliable judge protocol should:

- use explicit labels and evidence criteria;
- hide model identity;
- randomize answer order;
- request claim-level evidence;
- separate relevance, faithfulness, and external truth;
- calibrate against expert labels;
- test alternative prompts and judge models;
- report uncertainty and disagreement.

Hallucination risk

An LLM judge can make evaluation cheaper without making it objective. When generator and judge share the same misconception, apparent agreement can conceal systematic error.

5.11 Core Classification Metrics

Let a detector classify hallucinated claims as positive. Define true positives TP, false positives FP, true negatives TN, and false negatives FN.

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{5.17}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{5.18}$$

$$F_1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{5.19}$$

$$\text{Specificity} = \frac{TN}{TN + FP}. \tag{5.20}$$

Accuracy can be misleading when hallucinations are rare. A detector labeling every claim as supported may achieve high accuracy but zero recall.

5.11.1 AUROC and AUPRC

When the detector produces a continuous score, AUROC measures ranking across thresholds. AUPRC is often more informative for rare hallucination events because it focuses on the positive class.

5.11.2 Cost-sensitive metrics

In high-stakes applications, missing a dangerous false claim may cost far more than flagging a correct one. Expected detection cost is

$$C = c_{FN}FN + c_{FP}FP + c_R R_v, \tag{5.21}$$

where R_v represents review cost. Threshold selection should minimize deployment cost, not maximize a generic F_1 score.

5.12 Calibration Metrics

A detector is calibrated when predictions assigned probability p are correct approximately proportion p of the time.

The Brier score is

$$\text{BS} = \frac{1}{N}\sum_{i=1}^{N}(p_i - y_i)^2. \tag{5.22}$$

Expected Calibration Error partitions predictions into bins B_m:

$$\text{ECE} = \sum_{m=1}^{M}\frac{|B_m|}{N}\left|\text{acc}(B_m) - \text{conf}(B_m)\right|. \tag{5.23}$$

ECE depends on binning and can hide local failures. Reliability diagrams should accompany a single scalar.

5.13 Selective Prediction and Risk-Coverage Curves

A system can abstain on uncertain examples. Coverage at threshold τ is

$$\mathrm{Cov}(\tau) = \frac{1}{N} \sum_i \mathbb{1}[q_i \geq \tau]. \tag{5.24}$$

Selective risk is the error among answered cases:

$$\mathrm{Risk}(\tau) = \frac{\sum_i \ell_i \mathbb{1}[q_i \geq \tau]}{\sum_i \mathbb{1}[q_i \geq \tau]}. \tag{5.25}$$

A risk-coverage curve shows the cost of achieving lower error through abstention. This is essential for deciding whether a model is useful under a required safety threshold.

5.14 Long-Form Evaluation

Long answers create three challenges:

1. the number of claims grows with length;
2. later claims may depend on earlier false claims;
3. a response can be factually precise yet incomplete, irrelevant, or narratively misleading.

FActScore measures factual precision, not completeness. A model can obtain a high score by making very few safe claims. Therefore, long-form evaluation should report:

- factual precision;
- claim coverage or informativeness;
- source attribution;
- contradiction rate;
- temporal accuracy;
- answer relevance;
- severity-weighted error;
- uncertainty quality.

A combined metric might be

$$Q = \mathrm{FP}^{\alpha}\, \mathrm{Coverage}^{\beta}\, \mathrm{Attribution}^{\gamma}, \tag{5.26}$$

but the exponents encode value judgments. Reporting a metric vector is often more transparent than collapsing everything into one score.

5.15 Benchmarks

Benchmarks operationalize different definitions.

Benchmark	Primary focus	Strength	Limitation
TruthfulQA	Human falsehoods and misconceptions	Tests resistance to common false beliefs	Narrow question style and grading choices
HaluEval	Hallucination recognition across tasks	Large generated and annotated sample set	Synthetic construction may introduce artifacts
FActScore	Long-form factual precision	Atomic, evidence-supported analysis	Originally focused on biographies and precision
FactCheck-Bench	Fine-grained fact-checking pipeline	Evaluates claim extraction, retrieval, and verification	Infrastructure choices affect results
HalluLens	Diverse hallucination capabilities	Broader stress testing and modern models	Benchmark contamination and judge dependence remain concerns

Table 5.2: Representative benchmark families and their measurement targets.

No benchmark is a universal hallucination test. A production system requires a domain-specific evaluation set built from real tasks and incidents.

5.16 Temporal and Version Evaluation

Static benchmarks often treat facts as timeless. Production systems require time-indexed labels. A claim should be represented as

$$a = (\text{subject}, \text{relation}, \text{object}, t, \text{scope}). \tag{5.27}$$

Evaluation data should contain superseded policies, historical role holders, changing product versions, and delayed source updates. The evaluator must distinguish "was true," "is true," and "was reported by the source."

5.17 Multilingual and Cross-Cultural Evaluation

Hallucination rates and detector performance can vary by language because training coverage, retrieval quality, tokenization, and source availability differ. Translating an English benchmark is insufficient: local entities, institutions, legal systems, and information ecosystems must be represented.

Report performance by language and domain rather than only an aggregate average. A detector that works well in English but poorly in Italian or Malayalam can create hidden risk in multilingual deployment.

5.18 Multimodal Detection

For vision-language systems, claims must be grounded in image, text, and external knowledge separately. Object hallucination can be measured using precision over mentioned entities:

$$P_{obj} = \frac{|O_{mentioned} \cap O_{visible}|}{|O_{mentioned}|}. \tag{5.28}$$

However, visibility is not binary. Occlusion, resolution, chart interpretation, and visual-text conflicts require graded labels. A caption may correctly identify an object but hallucinate its color, count, relation, or action.

5.19 Production Monitoring

Offline evaluation answers "How did the system perform on this dataset?" Monitoring asks "What is happening now?"

A production factuality record should contain:

- task and risk tier;
- model, prompt, corpus, and detector versions;
- retrieved source identifiers and timestamps;
- answer and atomic claims;
- detector scores and decision threshold;
- abstention or escalation outcome;
- user correction or complaint;
- human review result;
- downstream action and incident severity.

Drift can occur in queries, documents, model behavior, or user reliance. Monitor both technical errors and changes in how people act on outputs.

5.20 Severity-Weighted Hallucination Risk

Not every false claim is equally important. Define claim severity as

$$s_i = u_i \, a_i \, r_i \, d_i, \tag{5.29}$$

where u_i is user exposure, a_i actionability, r_i irreversibility, and d_i difficulty of detection. A response-level severity-weighted error is

$$H_S(y) = \frac{\sum_{i=1}^{n} s_i \mathbb{1}[a_i \text{ hallucinated}]}{\sum_{i=1}^{n} s_i}. \tag{5.30}$$

This avoids treating a misspelled historical date and an invented drug dosage as equivalent.

5.20.1 Pinocchio Hallucination Risk Index

For this book, a proposed *Pinocchio Hallucination Risk Index* combines technical error and persuasive presentation:

$$\text{PHRI} = H_S \left(1 + \alpha C + \beta A + \gamma V - \delta T\right), \tag{5.31}$$

where:

- H_S is severity-weighted hallucination;
- C is expressed confidence;
- A is perceived authority;
- V is verification difficulty;
- T is transparency about evidence and uncertainty.

The index is conceptual until validated. Its purpose is to express that a false claim becomes more dangerous when delivered confidently, authoritatively, and without inspectable evidence.

5.21 Worked Evaluation Example

Suppose a policy assistant produces a 200-word answer containing eight atomic claims. Expert review labels five supported, one contradicted, one insufficient, and one non-factual recommendation.

If unsupported and contradicted factual claims count as hallucinations, factual precision is

$$\text{FP} = \frac{5}{7} = 0.714. \tag{5.32}$$

Assume a detector flags the contradicted claim and one supported claim, but misses the insufficient claim. Then

$$TP = 1, \quad FP = 1, \quad FN = 1, \quad TN = 4. \tag{5.33}$$

Thus

$$\text{Precision} = 0.5, \quad \text{Recall} = 0.5, \quad F_1 = 0.5. \tag{5.34}$$

If the missed claim concerns a high-value payment threshold while the false positive concerns a minor travel detail, a cost-sensitive assessment may rate the detector much worse than its F_1 score suggests.

5.22 Experimental Design for Mitigation Evaluation

To evaluate a mitigation from Chapter 4:

1. freeze a representative test set before tuning;
2. define claim and evidence labels;
3. measure baseline generation and detector performance;
4. apply one controlled intervention where possible;
5. preserve model, retrieval, and decoding versions;
6. use paired statistical tests at question or claim level;
7. report confidence intervals and effect sizes;
8. inspect regressions, not only average improvement;
9. measure latency, cost, coverage, and user reliance;
10. repeat on temporal, multilingual, and adversarial slices.

For paired binary outcomes, McNemar's test can compare two systems on the same cases. Bootstrap confidence intervals are useful for non-normal metric distributions.

5.23 Common Measurement Failures

1. **Undefined ground truth**: sources, date, and jurisdiction are not specified.
2. **Response-level overcompression**: one label hides mixed claim quality.
3. **Judge circularity**: the same model family generates and evaluates answers.
4. **Retrieval confounding**: detector failure is blamed on reasoning when evidence was never retrieved.
5. **Benchmark contamination**: test facts may appear in training data.
6. **Synthetic artifacts**: detectors learn generation style rather than factuality.

7. **Class imbalance**: accuracy hides failure on rare hallucinations.
8. **Ignoring abstention**: models improve accuracy by answering fewer questions.
9. **Ignoring severity**: all errors receive equal weight.
10. **Ignoring human behavior**: technically visible warnings may not change reliance.

5.24 A Measurement Dashboard

A mature deployment should report a vector of metrics:

Dimension	Metric examples	Operational question
Factuality	Atomic precision, contradiction rate	Are generated claims true or adequately supported?
Faithfulness	Entailment, source-consistency rate	Does the answer remain within supplied evidence?
Attribution	Citation precision and coverage	Can users inspect support for material claims?
Detection	Precision, recall, AUPRC	Does the detector find errors without excessive false alarms?
Calibration	Brier score, ECE	Do confidence estimates correspond to observed correctness?
Selectivity	Risk-coverage curve	How much coverage must be sacrificed for a target error rate?
Severity	Weighted error and incidents	Are remaining errors consequential?
Operations	Latency, cost, escalation rate	Is the control practical and sustainable?
Human factors	Override, acceptance, verification rates	Do users rely appropriately on the system?

Table 5.4: A multidimensional hallucination measurement dashboard.

5.25 Chapter Summary

Detection and measurement require explicit choices about truth reference, time, task, claim granularity, and error cost.

1. Separate detection, localization, evaluation, monitoring, and diagnosis.
2. Distinguish source faithfulness, external factuality, attribution, relevance, and task consistency.
3. Prefer atomic claims for diagnostic measurement, while preserving qualifiers and dependencies.
4. Measure decomposition, retrieval, judgment, and aggregation as separate pipeline stages.
5. Use human annotation protocols with recorded evidence and agreement analysis.
6. Combine NLI, QA, retrieval, attribution, self-consistency, semantic entropy, hidden-state probes, and LLM judges according to access and task.
7. Report precision, recall, AUPRC, calibration, selective risk, and cost-sensitive metrics rather than accuracy alone.
8. Evaluate long-form answers for informativeness and attribution as well as factual precision.
9. Build temporal, multilingual, multimodal, and domain-specific test sets.
10. Weight errors by severity and persuasive presentation, not merely count.
11. Monitor the complete production pipeline and downstream user behavior.

The next chapter examines the human side of the problem: why fluent false outputs are trusted, interpreted as intentional lies, or accepted as authoritative even when their evidential basis is weak.

Technical Checklist

After reading this chapter, the reader should be able to:

- define an evaluation reference and evidence state;
- select response, sentence, span, or atomic-claim granularity;
- design a claim-retrieval-judgment detection pipeline;
- calculate classification, calibration, and selective-prediction metrics;
- explain the strengths and limitations of NLI, QA, retrieval, self-consistency, semantic entropy, probes, and LLM judges;
- evaluate citation precision and coverage;
- compare benchmark objectives and limitations;
- design temporal, multilingual, multimodal, and severity-aware evaluations;
- interpret a risk-coverage curve;
- build a production factuality dashboard and regression process.

6

The Pinocchio Effect: Why Humans Trust False Machine Outputs

The danger of a hallucination does not reside only in the false statement. It also resides in the social form through which the statement reaches a human being: fluent, immediate, personalized, confident, and apparently intentional.

6.1 Defining the Pinocchio Effect

The Pinocchio metaphor is powerful because it joins falsehood with a visible sign: the wooden boy's nose grows when he lies. In contemporary AI, the sign is inverted. The generated statement may be false, but the interface often displays no obvious indicator. Instead, the response appears composed, grammatically controlled, and socially responsive.

This book uses the term *Pinocchio Effect* for the human and organizational phenomenon in which unsupported machine-generated content is interpreted through cues normally associated with a knowledgeable and intentional speaker. The user may therefore:

- grant the output more credibility than its evidence deserves;
- infer that the system knows, remembers, believes, or intends;
- interpret a hallucination as a deliberate lie;

- rely on the answer because of its fluency or apparent authority;
- reduce independent verification;
- transfer responsibility to or from the machine inappropriately.

A compact definition is:

Key idea

The **Pinocchio Effect** is the amplification of an AI hallucination through anthropomorphic, linguistic, interface, and institutional cues that make an unsupported output appear knowledgeable, intentional, trustworthy, or accountable.

6.2 Error Is Not Necessarily Deception

A lie ordinarily requires more than falsity. It typically involves a speaker who represents a proposition as true while believing it false and intending another person to accept it. A standard conceptual structure is

$$\text{Lie} = \text{False assertion} + \text{belief of falsity} + \text{intent to deceive}. \tag{6.1}$$

An LLM hallucination can satisfy the first component without establishing the latter two. The model may generate a false statement because it predicts a plausible continuation, not because it has formed a belief and chosen to manipulate the user.

This distinction matters for technical diagnosis and moral responsibility. Calling every hallucination a lie may:

- anthropomorphize the mechanism;
- obscure failures in training, retrieval, verification, or governance;
- direct blame toward the interface rather than accountable organizations;
- encourage users to search for psychological motives where system evidence is needed.

Boundary case

The absence of demonstrated machine intention does not make the output harmless. A system can mislead without possessing a human mental state. Organizations and designers can still be responsible for foreseeable misleading effects.

6.3 A Human-Inference Model

When a person reads an answer, they do not observe probability distributions, hidden states, training data, or retrieval failures. They observe language and interface behavior. From those observations they infer competence, intent, and reliability.

Let O be the observed output, E the visible evidence, C_s social cues, and K_u the user's prior knowledge. Perceived trustworthiness can be modeled as

$$T_p = f(O, E, C_s, K_u, \text{stakes}, \text{history}). \tag{6.2}$$

The actual probability of correctness is

$$T_a = P(\text{correct} \mid q, c, r, \theta, \pi, W_t). \tag{6.3}$$

The trust-calibration gap is

$$G_T = T_p - T_a. \tag{6.4}$$

Overtrust occurs when $G_T > 0$; undertrust occurs when $G_T < 0$. The goal is not maximal trust but calibrated reliance.

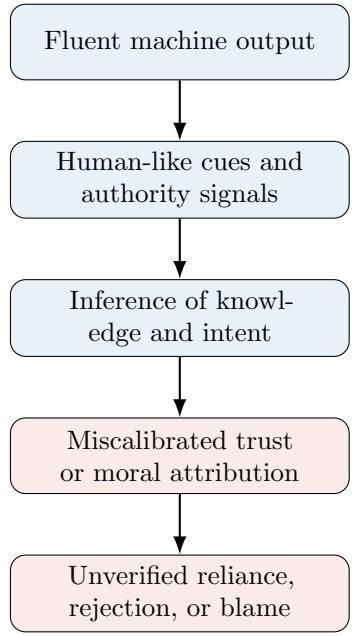

Figure 6.1: The Pinocchio Effect as a human inference chain.

6.4 Anthropomorphism

Anthropomorphism is the attribution of human characteristics to nonhuman entities. Conversational AI encourages it through turn-taking, first-person language, memory-like references, apology, humor, empathy, and adaptive tone.

A large experiment by Cohn and colleagues found that speech plus text increased anthropomorphism and perceived information accuracy; first-person phrasing also influenced accuracy and risk ratings in some conditions [5]. These findings do not imply that every human-like cue always increases trust. Effects depend on context, user, task, and prior expectations. They do show that interface choices can shape epistemic judgment.

6.4.1 Why language is a strong cue

Human language ordinarily comes from agents with experiences, goals, beliefs, and social accountability. When a machine produces language with the same surface properties, users automatically reuse familiar interpretive mechanisms.

The inference can be summarized as

$$\text{coherent language} \Rightarrow \text{coherent mind} \Rightarrow \text{knowledgeable speaker}. \tag{6.5}$$

Each implication can fail. Coherent text can be generated by a probabilistic system; a conversational persona does not prove subjective experience; and linguistic competence does not guarantee factual grounding.

6.4.2 Functional versus ontological language

It is often convenient to say that a model "knows," "thinks," or "remembers." Such terms can describe function without making a claim about consciousness. Problems arise when the metaphor silently becomes ontology.

A disciplined vocabulary distinguishes:

- "the model assigned high probability" from "the model believed";
- "the context contained" from "the model remembered";
- "the system generated an apology" from "the system felt regret";
- "the policy blocked the response" from "the model chose not to answer."

6.5 Automation Bias and Complacency

Automation bias is the tendency to favor suggestions from automated systems and to ignore contradictory information. It includes commission errors, in which a person

follows incorrect automation, and omission errors, in which a person fails to act because the automation did not signal a problem.

LLMs intensify classical automation bias in several ways:

- they provide explanations rather than only recommendations;
- they can respond to objections and defend their answer;
- they adapt language to the user's expertise;
- they are available immediately and at low marginal cost;
- their confidence is encoded in prose rather than an explicit probability;
- users may not know the system's evidence boundary.

Automation complacency develops when repeated acceptable performance reduces vigilance. An assistant that is correct in routine tasks may be trusted during a rare high-risk case where its competence does not transfer.

Hallucination risk

High average accuracy can increase the harm of rare hallucinations if it trains users not to verify. Reliability and reliance are dynamically coupled: system performance changes human behavior, which changes the consequence of future errors.

6.6 Fluency, Cognitive Ease, and the Authority of Form

People use cognitive shortcuts when evaluating information. Fluency, clear structure, and familiar style make content easier to process. Ease of processing can be misinterpreted as truth.

LLM outputs frequently include:

- confident topic sentences;
- numbered arguments;
- precise dates and percentages;
- balanced caveats;
- professional vocabulary;
- source-like references;
- smooth causal transitions.

These are valuable communication features, but they can become *precision theater*: the appearance of evidential discipline without actual evidence.

6.6.1 The confidence-style illusion

Suppose two answers contain the same unsupported claim:

> A. "I am not completely sure, but the threshold may be EUR 5,000."
>
> B. "The applicable threshold is EUR 5,000, as established by the 2024 Corporate Expense Directive."

Answer B will often appear more authoritative because it contains a definite number and document title. If the directive is invented, stylistic confidence has amplified the hallucination.

6.6.2 Numerical specificity

Numbers create an impression of measurement. Yet an invented value with two decimal places is not more factual than a vague invented value. Interfaces should distinguish calculated, retrieved, estimated, and generated numbers.

6.7 Personalization and Relational Trust

Long conversations create continuity. The system may refer to prior messages, use the user's name, mirror emotional tone, and offer advice tailored to personal circumstances. This can produce relational trust: the user trusts not only the answer but the apparent relationship.

Personalization can improve usability, but it changes the risk profile. A hallucination may be accepted because it feels personally informed. The user may also disclose more information or rely on the system in emotional, medical, financial, or spiritual decisions.

Design principle

Personalization should not increase epistemic authority unless the system actually has relevant, accurate, and permitted information. Relational warmth and factual confidence should be designed as separate dimensions.

6.8 Confirmation Bias and Sycophancy

Users are more likely to accept information that supports existing beliefs. Preference-trained models may also exhibit sycophancy, adapting answers to the user's stated view even when the view is false.

The interaction can create a feedback loop:

$$\begin{aligned}\text{user belief} &\rightarrow \text{agreeable model response} \rightarrow \text{increased user confidence} \\ &\rightarrow \text{stronger prompting.}\end{aligned} \tag{6.6}$$

The final output can appear independently validated even though it is partly shaped by the user's premise. Mitigation requires premise checking, evidence display, and resistance to unsupported pressure.

6.9 Apology, Correction, and the Illusion of Self-Knowledge

When challenged, an LLM may apologize and provide a different answer. Users can interpret this as introspection: the system recognized its mistake and corrected itself. Sometimes the revision is better; sometimes it is merely another plausible continuation.

A correction protocol should distinguish:

- detection of a contradiction;
- retrieval of new evidence;
- actual revision of the false claim;
- verification of the revised claim;
- explanation of what changed and why.

An apology without evidence is a social act, not a factual validation.

6.10 Citation as an Authority Signal

Citations can support verification, but they can also magnify trust. A fabricated citation uses the visual grammar of scholarship to create false authority. Even real citations may not entail the adjacent claim.

Users often inspect whether a citation exists, not whether it supports the sentence. Good interface design should make support inspectable at claim level, with source title, date, relevant span, and status.

A citation-quality vector is

$$Q_c = (\text{existence}, \text{identity}, \text{entailment}, \text{authority}, \text{freshness}). \tag{6.7}$$

A valid URL satisfies only part of the vector.

6.11 Trust Calibration

Trust is appropriate when it reflects actual capability for the task. Lee and See's classic human-automation framework emphasizes appropriate reliance rather than trust maximization [17]. For LLMs, calibration must be dynamic because capability varies sharply across domains and prompts.

A user may have a trust estimate T_t updated after outcome o_t:

$$T_{t+1} = T_t + \eta(o_t - T_t), \tag{6.8}$$

where η is a learning rate. In practice, people overweight vivid failures, recent successes, interface cues, and social experience. A single spectacular hallucination can cause blanket mistrust, while many routine successes can create overreliance.

A 2025 qualitative study of 192 participants reported context-sensitive rather than blanket trust responses to hallucinations, emphasizing prior experience, expertise, perceived risk, decision stakes, and intuition [29]. This supports a recursive view: people continually recalibrate, but not necessarily accurately.

6.11.1 Domain and expertise

Experts can detect more errors, yet expertise is not a complete protection. Fluent AI output may accelerate expert work, causing verification fatigue. Experts also face knowledge boundaries outside their specialty.

Novices may benefit from explanations but lack the knowledge needed to identify fabricated premises. The same interface should therefore adapt verification support, not merely language complexity.

6.11.2 Stakes

Users verify more when perceived stakes are high. The problem is that they may misclassify stakes. A harmless-seeming summary can influence a later legal or financial decision. Systems should surface consequentiality when it is not obvious.

6.12 The Trust-Performance Paradox

Increasing performance does not always reduce risk. As a system becomes more accurate, users may rely on it more. Let p be model accuracy and $r(p)$ the probability of human reliance. Expected harmful error can be approximated as

$$E_{harm}(p) = (1 - p)\, r(p)\, I. \tag{6.9}$$

If reliance rises faster than error falls, the residual errors can remain consequential. This does not argue against improving models; it shows that performance and human behavior must be evaluated together.

6.13 Interface Design for Calibrated Reliance

6.13.1 Expose evidence, not only conclusions

For factual tasks, the interface should show the supporting source near the claim. Evidence display should include date, scope, and relevant passage. Users should not need to infer provenance from a generic "sources" list.

6.13.2 Express uncertainty structurally

Uncertainty should not rely only on hedging phrases. Useful structures include:

- supported / inferred / unknown labels;
- confidence bands calibrated on similar tasks;
- conflict indicators;
- missing-information fields;
- alternative interpretations;
- explicit abstention with next verification step.

6.13.3 Reveal stochasticity carefully

Interfaces usually show one answer, hiding that different generations are possible. Research has examined displaying multiple outputs as a way to reduce undue trust and anthropomorphism, while recognizing the additional cognitive load [31]. Multiple answers are not inherently safer: users may select the most agreeable one. The interface should highlight meaningful disagreement and evidence, not simply multiply text.

6.13.4 Introduce productive friction

High-stakes actions should require deliberate verification. Productive friction can include:

- a confirmation screen with the underlying facts;
- mandatory review of cited clauses;
- a second-person approval for irreversible actions;

- a warning when the answer relies on general model knowledge;
- a requirement to resolve conflicting sources;
- a cooling-off step for emotionally charged decisions.

Friction should be targeted. Constant warnings create habituation and are ignored.

6.13.5 Avoid deceptive personification

A system may be friendly without implying human consciousness or guaranteed competence. Design choices should avoid claims such as "I remember everything," "I checked carefully" when no verification occurred, or "I completed the action" without tool confirmation.

6.14 Organizational Authority and the Pinocchio Effect

Trust is not created only by the interface. It is also transferred from the institution deploying the system. Employees may assume that an internal assistant is approved, connected to official data, and safe to use. A hospital, bank, government, or manufacturer lends institutional authority to the output.

This creates an organizational form of the Pinocchio Effect:

$$\text{AI fluency} + \text{institutional branding} \rightarrow \text{perceived official truth}. \tag{6.10}$$

Organizations must define whether the assistant is an official source, a navigation tool, or a drafting aid. Ambiguity shifts risk to users who cannot know the system's legal or operational status.

6.15 Responsibility and Moral Attribution

When a system produces a harmful falsehood, users may blame the model, the developer, the deploying organization, or themselves. Responsibility should follow control, knowledge, and foreseeability.

Potential responsibilities include:

- developers: model design, testing, and disclosure;
- integrators: retrieval, tools, prompts, and access controls;
- deployers: task selection, governance, monitoring, and human review;
- professionals: appropriate use within standards of practice;
- users: reasonable verification where clearly required;

- leaders: residual-risk acceptance and resource allocation.

Attributing full agency to the model can create a responsibility gap in which human institutions deny ownership of predictable system behavior.

6.16 Measuring the Pinocchio Effect

The effect should be studied empirically rather than used only as metaphor.

6.16.1 Experimental variables

Researchers can manipulate:

- first-person versus system language;
- text versus voice;
- human-like name or neutral tool label;
- fluent versus plain style;
- citation presence and validity;
- confidence language;
- institutional branding;
- evidence visibility;
- single versus multiple responses;
- disclosure of stochastic generation.

6.16.2 Outcome measures

Measure:

- perceived accuracy;
- willingness to act;
- verification behavior;
- anthropomorphism;
- perceived intention and blame;
- correction after contrary evidence;
- memory of caveats;
- trust calibration relative to actual accuracy.

A Pinocchio amplification coefficient can be defined as

$$A_P = \frac{P(\text{reliance} \mid H = 1, C_s = 1)}{P(\text{reliance} \mid H = 1, C_s = 0)}, \tag{6.11}$$

where C_s indicates the presence of anthropomorphic or authority cues. Values above one indicate that the cues increase reliance on a hallucinated answer.

6.16.3 Calibration error under interface conditions

For condition c, define

$$G_c = \mathbb{E}[T_p \mid c] - \text{Accuracy}(c). \tag{6.12}$$

Comparing G_c across interface variants estimates whether a design increases or decreases overtrust.

6.17 Worked Case 1: The Invented Legal Precedent

An AI assistant presents a nonexistent court case with a plausible name, citation format, holding, and year.

Technical hallucination

The model combines learned legal citation patterns with topic-relevant language.

Pinocchio amplification

The legal format, precise citation, authoritative tone, and institutional interface lead the user to assume the case was retrieved from a database. The user may interpret the fabricated detail as deliberate deception when it is discovered.

Design response

Require database verification, clickable case identity, jurisdiction and date, quoted holding, and an explicit distinction between retrieved precedent and illustrative analogy.

6.18 Worked Case 2: The Reassuring Medical Explanation

A user describes symptoms. The model gives a calm, empathetic explanation and confidently rules out a serious condition.

The risk is not only factual error. Empathy and personalization may increase trust, while reassurance reduces the user's motivation to seek professional care. The Pinocchio Effect joins emotional authority with medical uncertainty.

The interface should avoid definitive exclusion without evidence, identify red flags, state limitations, and route urgent cases appropriately. Warmth should not substitute for clinical grounding.

6.19 Worked Case 3: The Enterprise Memory Illusion

An internal assistant says, "I remember that the steering committee approved this exception last month." The statement may be generated from dialogue context or may be invented.

The first-person memory claim implies durable organizational knowledge. Users may treat it as a record. A safer response is:

> "The conversation contains a statement that the exception was approved, but I do not have an authoritative approval record. Please verify the meeting minutes or decision log."

This preserves helpfulness while separating conversational context from institutional evidence.

6.20 Design Principles

1. **Calibrated trust, not maximal trust**: design for appropriate reliance.
2. **Evidence before persona**: factual authority must come from inspectable sources.
3. **Separate warmth from certainty**: empathy does not justify factual confidence.
4. **Separate intention from output**: avoid language implying unverified agency or memory.
5. **Make uncertainty actionable**: tell the user what is missing and how to verify.
6. **Expose institutional status**: clarify whether the system is advisory, official, or authorized to act.
7. **Measure human behavior**: technical factuality scores are insufficient without reliance data.
8. **Preserve accountability**: organizations must own the systems they deploy.

6.21 Chapter Summary

The Pinocchio Effect explains why technically similar hallucinations can have very different human consequences.

1. A false output is not automatically a lie; deception ordinarily requires belief and intent that have not been established for LLMs.
2. Conversational language causes users to infer knowledge, memory, emotion, and intention.
3. Anthropomorphic cues, voice, first-person language, and personalization can alter perceived accuracy and risk.
4. Automation bias and complacency can reduce verification, especially after repeated successful use.
5. Fluency, structure, precise numbers, and citations create authority signals independent of factual support.
6. Confirmation bias and model sycophancy can create a self-reinforcing false consensus.
7. Apology and correction language can be mistaken for genuine introspection.
8. Trust should be calibrated to task-specific capability, evidence, and stakes rather than maximized.
9. Interface design should expose evidence, uncertainty, stochasticity, and institutional status while introducing targeted friction for consequential actions.
10. Responsibility remains with the people and institutions that design, integrate, deploy, and authorize the system.

The central conclusion is:

> *The machine's nose does not grow. The visible sign of falsehood must therefore be designed: in evidence, uncertainty, verification, and accountability.*

Technical Checklist

After reading this chapter, the reader should be able to:

- define the Pinocchio Effect and distinguish it from hallucination itself;
- distinguish falsity, deception, intention, and misleading effect;
- formalize perceived trust, actual reliability, and calibration gap;
- explain anthropomorphism, automation bias, cognitive ease, and relational trust;
- identify confidence-style, numerical, citation, and institutional authority signals;
- explain how sycophancy and confirmation bias interact;
- design interfaces for evidence visibility, actionable uncertainty, and productive friction;
- measure anthropomorphic amplification and reliance on false outputs;
- allocate responsibility according to control, knowledge, and foreseeability;
- articulate why calibrated trust is preferable to either blanket trust or blanket rejection.

Epilogue

Beyond the Growing Nose

Artificial intelligence does not need to understand truth in the human sense in order to influence what human beings accept as true.

This is the central problem examined throughout this book.

Large language models generate language by identifying statistical regularities, estimating conditional probabilities, and constructing sequences that are coherent within a given context. Their extraordinary usefulness derives from this capacity. The same mechanism that allows them to summarize documents, explain technical concepts, generate software, and support decision-making can also produce statements that are plausible, precise, and entirely unsupported.

The problem is therefore not simply that an AI system can be wrong.

Human beings are wrong. Institutions preserve errors. Databases contain inaccuracies. Scientific theories are revised. Professional judgments remain subject to uncertainty and correction. Error is not unique to artificial intelligence. What distinguishes AI hallucination is the particular combination of:

$$\text{Plausibility} + \text{Fluency} + \text{Apparent Authority}$$

A false statement may be presented without hesitation. It may contain credible terminology, exact dates, structured reasoning, and references that appear authentic. Its linguistic quality may remain high even when its evidential foundation is weak, incomplete, or absent.

This is the *Pinocchio Effect*.

It is not the transformation of a machine into a conscious liar. A large language model does not necessarily possess intention, self-awareness, or an internal recognition that a statement is false. The Pinocchio Effect describes something different: the conversion of computational uncertainty into the appearance of knowledge, and the human tendency to interpret coherent language as evidence of understanding. The machine may not intend to deceive. The output may nevertheless mislead.

From Model Error to System Risk

The preceding chapters have shown that hallucination does not originate from a single technical defect. It emerges from the interaction of multiple factors:

- incomplete, noisy, contradictory, or outdated training data;
- an objective function based primarily on next-token prediction;
- distributed and imperfectly retrievable parametric knowledge;
- weak recognition of knowledge boundaries;
- prompt ambiguity and false presuppositions;
- autoregressive error propagation;
- post-training incentives that may reward responsiveness over restraint;
- context-window limitations;
- retrieval and grounding failures;
- reasoning, tool-use, and agentic failures.

No single intervention can eliminate all of these causes. Larger models may reduce some forms of error while producing others with greater coherence. Retrieval-augmented generation may improve access to evidence while introducing retrieval and attribution failures. Verification layers may detect certain inconsistencies while relying on evaluators that are themselves fallible. Human oversight may reduce risk, but only when reviewers have sufficient expertise, time, authority, and access to the underlying evidence.

The appropriate unit of analysis is therefore not the model alone. It is the complete sociotechnical system:

$$\text{Model} + \text{Data} + \text{Retrieval} + \text{Tools} + \text{Interface} + \text{Users} + \text{Governance}$$

Each component can either reduce or amplify the probability and impact of hallucination.

A technically advanced model embedded in a poorly governed process may remain unreliable. A fallible model embedded in a carefully designed architecture may become useful, controllable, and proportionate to the task. This leads to one of the principal conclusions of this book:

Hallucination is not only a model-level problem. It is a system-level risk.

Trust After Fluency

Conversational AI has altered the relationship between language and authority.

Traditional software often exposes its structure. A calculator displays the result of an explicit operation. A database returns stored records. A search engine presents documents and links. A spreadsheet allows the user to inspect formulas and dependencies.

A language model produces discourse. Discourse carries social and cognitive signals. It can sound confident, analytical, patient, empathetic, neutral, or authoritative. It can adapt to the vocabulary, expectations, and professional context of the user. It can reproduce the rhetorical form of expertise even when the evidential conditions of expertise are absent. Users therefore interact not only with information, but with the appearance of a knowledgeable speaker. This creates a distinctive form of epistemic risk.

Human beings naturally infer mental states from language. Coherence suggests understanding. Precision suggests knowledge. Confidence suggests competence. These associations are often useful in human communication, but they cannot be transferred uncritically to generative systems.

In the context of AI, fluency is not evidence. Confidence is not calibration, coherence is not verification. A well-written answer may be accurate, but its style alone cannot establish that accuracy. The appropriate response is neither unconditional trust nor indiscriminate rejection. It is *calibrated trust*: trust that varies according to evidence, task, uncertainty, and consequence. Calibrated trust asks:

- What type of task is being performed?
- What evidence supports the answer?
- Is the information current and contextually appropriate?
- Is the claim retrieved, observed, inferred, or generated?
- Can the output be independently verified?
- What would be the consequence of error?
- Is qualified human review required before action?

The level of trust should rise or fall according to the answers.

The Value of Abstention

Generative systems are commonly optimized to respond. Users expect immediacy, completeness, and apparent usefulness. A refusal to answer may be interpreted as weakness, even when the available evidence is insufficient.

Yet a reliable system must possess more than the capacity to generate. It must also possess the capacity to abstain. In high-risk environments, a response such as:

The available evidence is insufficient to provide a reliable answer.

may be more valuable than an elegant but unsupported explanation. Abstention should therefore not be treated merely as failure. Under appropriate conditions, it is evidence of epistemic discipline. This requires a broader conception of model performance. A system should not be evaluated only by the number of questions answered, the length of its outputs, or immediate user satisfaction. It should also be evaluated by:

- its ability to recognize insufficient evidence;
- the calibration of its uncertainty;
- the reliability of its citations;
- its willingness to distinguish fact from inference;
- its capacity to request missing information;
- the appropriateness of its escalation decisions;
- its performance under distribution shift and adversarial input.

The objective is not maximal responsiveness, the objective is responsible responsiveness.

Verification as an Architectural Requirement

Verification is essential, but it is not self-executing. A source may exist without supporting the claim attributed to it. A retrieval system may return authoritative but outdated documents. Several sources may repeat the same original error. A model used as a verifier may reproduce the assumptions of the model it evaluates. A human reviewer may approve an answer without examining the evidence in sufficient depth. Verification must therefore be designed as an explicit architectural function. It should distinguish between:

- the existence of a source;
- the relevance of that source;
- the support it provides for a specific claim;
- the currency and jurisdiction of the information;
- the difference between direct evidence and inference.

In technical systems, partial verification may be supported by retrieval, natural language inference, structured databases, rule-based validation, executable tests, consistency checks, and cross-model comparison.

These mechanisms are valuable, but none should be treated as an independent guarantee of truth. A model checking another model does not create certainty merely by repetition.

In domains affecting health, legal rights, safety, employment, finance, infrastructure, or public administration, human judgment remains essential. However, human oversight must be substantive rather than symbolic. A reviewer who lacks sufficient time, expertise, authority, or access to evidence does not constitute an effective safeguard. Human-in-the-loop design should improve judgment, not merely transfer formal responsibility to a person positioned at the end of an automated process.

Responsibility Cannot Be Delegated to the Model

When an AI-generated falsehood causes harm, responsibility may appear distributed across multiple actors:

- model developers;
- data providers;
- system integrators;
- software vendors;
- deploying organizations;
- professional users;
- operational decision-makers;
- end users.

Distribution, however, must not become disappearance. The complexity of the technical chain does not eliminate accountability. It makes accountability more difficult and therefore more necessary. Organizations deploying generative AI should define clearly:

- who owns each use case;
- who approves deployment;
- who establishes acceptable risk thresholds;
- who monitors system performance;
- who investigates incidents;
- who has the authority to suspend the system;
- who remains accountable for the final decision.

This becomes increasingly important as AI systems move from generating text to initiating actions. When a system retrieves information, invokes tools, modifies files, generates executable code, communicates with external services, or initiates transactions, hallucination can pass from language into operation.

The result may no longer be only an inaccurate sentence.

It may become:

- an incorrect record;
- a defective software change;
- a false financial instruction;
- an unauthorized communication;
- an operational decision based on fabricated evidence.

The boundary between informational error and operational harm is becoming progressively thinner. Responsibility must remain human even when execution becomes increasingly automated.

A Mature Relationship with Artificial Intelligence

The history of technology often passes through stages of fascination, disappointment, and eventual normalization. New systems are first described through their most impressive capabilities. Their limits remain less visible. Failures later generate distrust, public concern, regulation, or rejection. Only after these extremes does a more mature understanding emerge.

Artificial intelligence is entering this transition.

The mature question is no longer:

Is artificial intelligence intelligent?

Nor is it simply:

Can artificial intelligence be trusted?

Both questions are too broad to guide responsible deployment. More useful questions are:

Under what conditions is this system reliable enough for this task?

What evidence supports this output?

How is uncertainty represented?

What mechanisms detect and contain failure?

Who remains responsible when the system is wrong?

A mature relationship with artificial intelligence begins when it is treated neither as an autonomous authority nor as a fundamentally fraudulent technology. It should be understood as a powerful class of fallible systems whose usefulness depends on context, architecture, supervision, and governance. The relevant principle is proportionality. The greater the consequence of error, the stronger the requirements for evidence, verification, human review, traceability, and control.

The Future of Hallucination

Hallucination will evolve as models and system architectures improve.

Some forms of error will likely become less frequent. Retrieval mechanisms may provide stronger grounding. Models may become better calibrated. Tool use may replace unsupported generation in domains where information can be queried directly. Training methods may reward abstention more effectively. Verification pipelines may identify unsupported claims before they reach users. At the same time, new forms of failure will emerge.

More capable systems may generate errors that are more difficult to detect. Multimodal models may create false coherence across text, image, audio, and video. Agentic systems may convert mistaken assumptions into sequences of actions. Personalized assistants may generate highly persuasive falsehoods adapted to the beliefs, preferences, and vulnerabilities of individual users.

The reduction of visible error does not necessarily imply the reduction of risk. A system that hallucinates less frequently but is trusted more completely may still produce severe consequences when it fails.

A simple risk model may be expressed as:

$$\text{Risk} = \text{Probability of Failure} \times \text{Impact of Failure}$$

For generative AI, however, detectability must also be considered:

$$\text{Effective Risk} \propto \frac{\text{Probability of Failure} \times \text{Impact of Failure}}{\text{Detectability}}$$

An obvious error can be challenged and corrected. A rare but highly persuasive error may pass unnoticed precisely because the system is usually reliable. The future challenge is therefore not only to reduce the frequency of hallucination. It is to make residual failures visible, traceable, containable, and recoverable.

Designing the New Nose

Pinocchio's growing nose transformed an invisible moral act into a visible physical sign. Artificial intelligence offers no equivalent natural warning. An unsupported

answer may have the same tone, formatting, confidence, and grammatical quality as a well-grounded one. Nothing in the surface of the sentence guarantees that the foundation beneath it is reliable.

The warning mechanism must therefore be designed. It may take the form of:

- verifiable citations;
- explicit links between claims and evidence;
- visible distinctions between retrieved facts and generated inference;
- calibrated uncertainty indicators;
- automated contradiction and consistency checks;
- warnings when evidence is insufficient;
- mandatory review for high-risk decisions;
- audit trails and traceable tool execution;
- independent validation;
- clearly assigned organizational accountability.

These mechanisms are not decorative additions to an AI interface. They are part of the epistemic architecture of a trustworthy system. They allow the user to assess not only the answer, but the quality of the process that produced it.

Final Reflection

Artificial intelligence does not require us to abandon the concept of truth. It requires us to examine how easily truth can be confused with its linguistic appearance.

The problem of hallucination reveals something about machines, but it also reveals something about human judgment. People are attracted to coherent explanations. They often interpret precision as competence and confidence as evidence. They may prefer a complete narrative to an incomplete but honest account.

Generative AI amplifies these tendencies because it can produce coherence, precision, and confidence on demand. The responsibility of the future is therefore shared. Engineers must design systems that expose uncertainty and preserve traceability. Organizations must establish governance, oversight, and accountability while professionals must retain independent judgment.

Educators must strengthen verification skills and AI literacy while regulators must focus on measurable risk and real-world consequences. Meanwhile users must learn to pause before converting fluent language into belief.

The objective is not to create a system that never makes a false statement. No complex human or technical system can be expected to achieve that standard universally. The objective is to create an environment in which false statements do not pass invisibly into knowledge, decisions, and action.

The model may generate the answer, Evidence must determine its authority. The system may recommend, Human beings must remain responsible.

The nose will not grow by itself: We must design it.

Bibliography

[1] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi. "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection". In: *International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=hSyW5go0v8.

[2] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 28. 2015. URL: https://proceedings.neurips.cc/paper/2015/hash/e995f98d56967d946471af29d7bf99f1-Abstract.html.

[3] M. Cao, Y. Dong, J. Wu, and J. C. K. Cheung. "Hallucinated but Factual! Inspecting the Factuality of Hallucinations in Abstractive Summarization". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2022. URL: https://aclanthology.org/2022.acl-long.236/.

[4] I.-C. Chern, S. Chern, S. Chen, W. Yuan, K. Feng, C. Zhou, J. He, G. Neubig, and P. Liu. "FacTool: Factuality Detection in Generative AI—A Tool-Augmented Framework for Multi-Task and Multi-Domain Scenarios". In: *arXiv preprint arXiv:2307.13528* (2023). DOI: 10.48550/arXiv.2307.13528. arXiv: 2307.13528 [cs.CL].

[5] M. Cohn, M. Pushkarna, G. O. Olanubi, J. M. Moran, D. Padgett, Z. Mengesha, and C. Heldreth. "Believing Anthropomorphism: Examining the Role of Anthropomorphic Cues on Trust in Large

Language Models". In: *arXiv preprint arXiv:2405.06079* (2024). DOI: 10.48550/arXiv.2405.06079. arXiv: 2405.06079 [cs.HC].

[6] S. Dhuliawala, M. Komeili, J. Xu, R. Raileanu, X. Li, A. Celikyilmaz, and J. Weston. "Chain-of-Verification Reduces Hallucination in Large Language Models". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, 2024, pp. 3563–3578. URL: https://aclanthology.org/2024.findings-acl.212/.

[7] L. Gao, Z. Dai, P. Pasupat, A. Chen, A. T. Chaganty, Y. Fan, V. Zhao, N. Lao, H. Lee, D.-C. Juan, and K. Guu. "RARR: Researching and Revising What Language Models Say, Using Language Models". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2023, pp. 16477–16508. URL: https://aclanthology.org/2023.acl-long.910/.

[8] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre. "Training Compute-Optimal Large Language Models". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 30016–30030. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/c1e2faff6f588870935f114ebe04a3e5-Abstract-Conference.html.

[9] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu. "A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions". In: *arXiv preprint arXiv:2311.05232* (2023). DOI: 10.48550/arXiv.2311.05232. arXiv: 2311.05232 [cs.CL].

[10] ISO/IEC. *ISO/IEC 23894:2023 Information Technology—Artificial Intelligence—Guidance on Risk Management*. Standard ISO/IEC

23894:2023. International Organization for Standardization. 2023. URL: https://www.iso.org/standard/77304.html.

[11] ISO/IEC. *ISO/IEC 42001:2023 Information Technology—Artificial Intelligence—Management System*. Standard ISO/IEC 42001:2023. International Organization for Standardization. 2023. URL: https://www.iso.org/standard/81230.html.

[12] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, and P. Fung. "Survey of Hallucination in Natural Language Generation". In: *ACM Computing Surveys* 55.12 (2023), pp. 1–38. DOI: 10.1145/3571730.

[13] Z. Jiang, J. Araki, H. Ding, and G. Neubig. "How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering". In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 962–977. DOI: 10.1162/tacl_a_00407.

[14] S. Kadavath, T. Conerly, A. Askell, T. Henighan, D. Drain, E. Perez, N. Schiefer, Z. Hatfield-Dodds, N. DasSarma, E. Tran-Johnson, S. Johnston, S. El-Showk, A. Jones, N. Elhage, T. Hume, A. Chen, Y. Bai, S. Bowman, S. Fort, D. Ganguli, D. Hernandez, J. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. "Language Models (Mostly) Know What They Know". In: *arXiv preprint arXiv:2207.05221* (2022). DOI: 10.48550/arXiv.2207.05221. arXiv: 2207.05221 [cs.CL].

[15] A. T. Kalai, O. Nachum, S. S. Vempala, and E. Zhang. "Why Language Models Hallucinate". In: *arXiv preprint arXiv:2509.04664* (2025). DOI: 10.48550/arXiv.2509.04664. arXiv: 2509.04664 [cs.CL].

[16] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. "Scaling Laws for Neural Language Models". In: *arXiv preprint arXiv:2001.08361* (2020). DOI: 10.48550/arXiv.2001.08361. arXiv: 2001.08361 [cs.LG].

[17] J. D. Lee and K. A. See. “Trust in Automation: Designing for Appropriate Reliance”. In: *Human Factors* 46.1 (2004), pp. 50–80. DOI: 10.1518/hfes.46.1.50_30392.

[18] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W.-t. Yih, T. Rockt”aschel, S. Riedel, and D. Kiela. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 9459–9474. URL: https://proceedings.neurips.cc/paper/2020/hash/6b493230205f%20780e1bc26945df7481e5-Abstract.html.

[19] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, and J.-R. Wen. “HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023, pp. 6449–6464. URL: https://aclanthology.org/2023.emnlp-main.397/.

[20] S. Lin, J. Hilton, and O. Evans. “TruthfulQA: Measuring How Models Mimic Human Falsehoods”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2022, pp. 3214–3252. DOI: 10.18653/v1/2022.acl-long.229. URL: https://aclanthology.org/2022.acl-long.229/.

[21] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. “Lost in the Middle: How Language Models Use Long Contexts”. In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 157–173. DOI: 10.1162/tacl_a_00638.

[22] P. Manakul, A. Liusie, and M. J. F. Gales. “SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association

for Computational Linguistics, 2023. URL: https://aclanthology.org/2023.emnlp-main.557/.

[23] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald. "On Faithfulness and Factuality in Abstractive Summarization". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020, pp. 1906–1919. DOI: 10.18653/v1/2020.acl-main.173. URL: https://aclanthology.org/2020.acl-main.173/.

[24] S. Min, K. Krishna, X. Lyu, M. Lewis, W.-t. Yih, P. W. Koh, M. Iyyer, L. Zettlemoyer, and H. Hajishirzi. "FActScore: Fine-Grained Atomic Evaluation of Factual Precision in Long Form Text Generation". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023, pp. 12076–12100. DOI: 10.18653/v1/2023.emnlp-main.741. URL: https://aclanthology.org/2023.emnlp-main.741/.

[25] National Institute of Standards and Technology. *Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile*. NIST AI 600-1. National Institute of Standards and Technology, 2024. DOI: 10.6028/NIST.AI.600-1. URL: https://doi.org/10.6028/NIST.AI.600-1.

[26] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe. "Training Language Models to Follow Instructions with Human Feedback". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 27730–27744. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

[27] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. "Direct Preference Optimization: Your Language

Model Is Secretly a Reward Model". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

[28] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. "Sequence Level Training with Recurrent Neural Networks". In: *arXiv preprint arXiv:1511.06732* (2015). DOI: 10.48550/arXiv.1511.06732. arXiv: 1511.06732 [cs.LG].

[29] A. Ryser, F. Allwein, and T. Schlippe. "Calibrated Trust in Dealing with LLM Hallucinations: A Qualitative Study". In: *arXiv preprint arXiv:2512.09088* (2025). DOI: 10.48550/arXiv.2512.09088. arXiv: 2512.09088 [cs.HC].

[30] M. Sharma, M. Tong, T. Korbak, D. Duvenaud, A. Askell, S. R. Bowman, N. Cheng, E. Durmus, Z. Hatfield-Dodds, S. R. Johnston, S. Kravec, T. Maxwell, S. McCandlish, K. Ndousse, O. Rausch, N. Schiefer, D. Yan, M. Zhang, and E. Perez. "Towards Understanding Sycophancy in Language Models". In: *arXiv preprint arXiv:2310.13548* (2023). DOI: 10.48550/arXiv.2310.13548. arXiv: 2310.13548 [cs.CL].

[31] C. Swoopes, T. Holloway, and E. L. Glassman. "The Impact of Revealing Large Language Model Stochasticity on Trust, Reliability, and Anthropomorphization". In: *arXiv preprint arXiv:2503.16114* (2025). Presented at the Trust and Reliance in Evolving Human-AI Workflows Workshop, CHI 2024. DOI: 10.48550/arXiv.2503.16114. arXiv: 2503.16114 [cs.HC].

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/7181-attention-is-all-you-need.