processes

MDPI

Article

Neuro-Symbolic Verification for Preventing LLM Hallucinations in Process Control

Boris Galitsky [1,*] **and Alexander Rybalov** [2]

[1] Knowledge Trail Inc., San Jose, CA 95127, USA
[2] LAMBDA Laboratory, Tel Aviv University, Ramat Aviv, Tel Aviv 6997801, Israel; alxndr_r@yahoo.com
* Correspondence: bgalitsky@hotmail.com

Abstract

Large Language Models (LLMs) are increasingly used in industrial monitoring and decision support, yet they remain prone to process-control hallucinations—diagnoses and explanations that sound plausible but conflict with physical constraints, sensor data, or plant dynamics. This paper investigates hallucination as a failure of abductive reasoning, where missing premises, weak mechanistic support, or counter-evidence lead an LLM to propose incorrect causal narratives for faults such as pump restriction, valve stiction, fouling, or reactor runaway. We develop a neuro-symbolic framework in which Abductive Logic Programming (ALP) evaluates the coherence of model-generated explanations, counter-abduction generates rival hypotheses that test whether the explanation can be defeated, and Discourse-weighted ALP (D-ALP) incorporates nucleus–satellite structure from operator notes and alarm logs to weight competing explanations. Using our 500-scenario Process-Control Hallucination Dataset, we assess LLM reasoning across mechanistic, evidential, and contrastive dimensions. Results show that abductive and counter-abductive operators substantially reduce explanation-level hallucinations and improve alignment with physical process behavior, particularly in "easy-but-wrong" cases where a superficially attractive explanation contradicts historian trends or counter-evidence. These findings demonstrate that abductive reasoning provides a practical and verifiable foundation for improving LLM reliability in safety-critical process-control environments.

Keywords: abductive reasoning; process-control hallucinations; counter-abduction; discourse-weighted ALP; fault diagnosis; neuro-symbolic LLM verification

Check for updates

Academic Editors: Sheng Du, Li Jin, Pan Yu and Hao Liu

Received: 11 December 2025
Revised: 9 January 2026
Accepted: 13 January 2026
Published: 16 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Large Language Models (LLMs) have recently been explored not only as conversational systems, but increasingly as autonomous decision-making components in process automation and industrial control. Their ability to parse technical language, generate control strategies, and interface with operations via natural language instructions makes them attractive as supervisory agents for tasks such as batch sequencing, alarm response, optimization, and procedure generation. However, this promise is constrained by a persistent and under-studied failure mode: hallucinated control commands—outputs that appear syntactically valid, operationally confident, and often persuasive, yet are physically infeasible, unsafe, or inconsistent with the underlying plant state.

Process systems engineering (PSE) is a core discipline in chemical engineering and is dedicated to the systematic design, modeling, optimization, and control of industrial processes. Traditionally, PSE has relied on mechanistic modeling approaches, in which

complex physical and chemical processes are mathematically formalized to predict system behavior under varying operating conditions [1,2]. Over time, the field has progressively incorporated data-driven methodologies, including optimization, estimation, and advanced control strategies, to improve decision-making under uncertainty [3]. PSE underpins critical industrial sectors such as chemical manufacturing, energy systems, and wastewater treatment, where efficiency, safety, cost reduction, and sustainability are paramount [4]. With the advent of Industry 4.0, digital technologies—particularly AI, large-scale data analytics, and automation—have accelerated a shift from purely physics-based models toward hybrid and learning-based approaches that support predictive analytics and real-time decision assistance [5].

Recently, LLMs have emerged as a promising interface layer for PSE, enabling natural-language interaction with process data, automated explanation generation, and decision support across monitoring, diagnosis, and control tasks. The integration of LLMs into PSE revolutionizes process modeling, control, and optimization by enhancing automation, decision-making, and knowledge extraction. By enhancing process efficiency, operational reliability, and resource optimization, LLM-enabled PSE contributes directly to industrial sustainability goals and broader societal benefits. The role of LLMs in key research domains of PSE, including data availability, process modeling, process control, and process optimization, is shown in Figure 1.

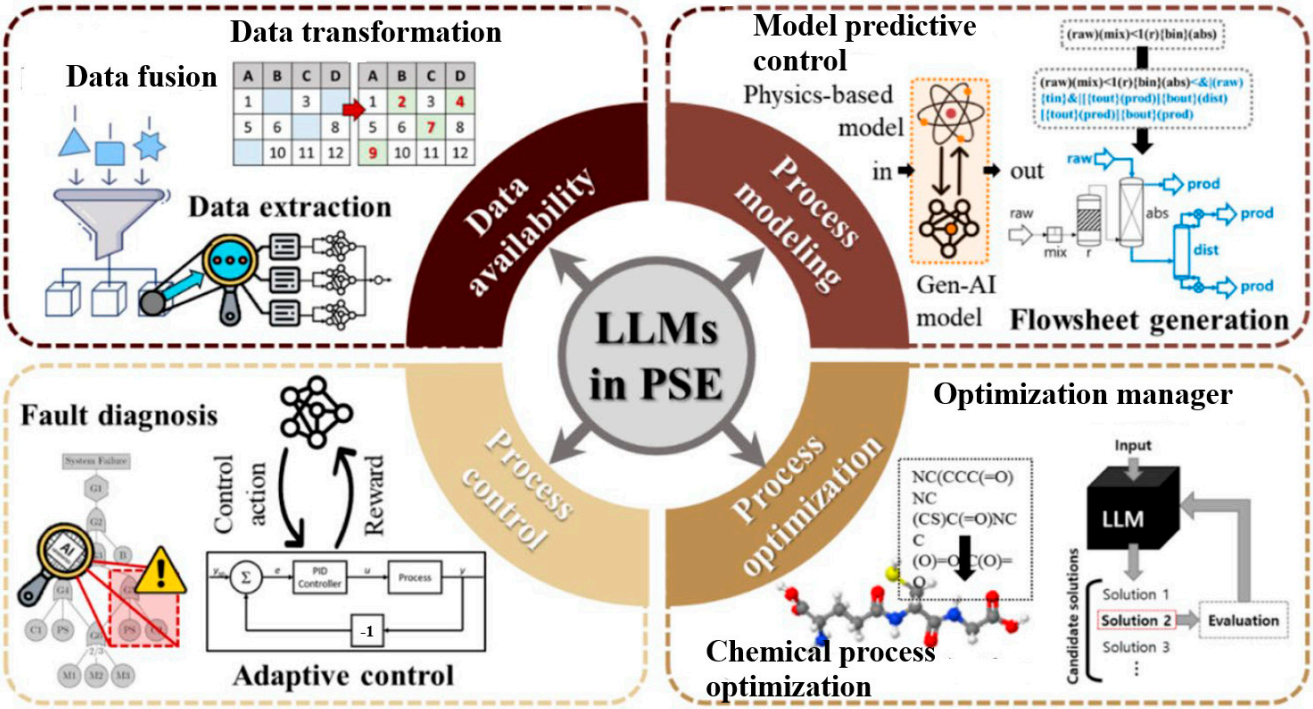


Figure 1. The role of LLMs in key research domains of process systems engineering.

The expanding role of LLMs across the PSE lifecycle are summarized—from data extraction and transformation, through hybrid physics–AI process modeling and flowsheet generation, to fault diagnosis, adaptive control, and process optimization. While this integration enables natural-language interfaces, automated modeling, and decision support under Industry 4.0, it also highlights where hallucinations can arise: during data interpretation, explanation of process behavior, fault attribution, and optimization recommendations. The chart motivates the need for verification mechanisms in LLM-enabled PSE, as unsupported or fluent-but-incorrect outputs at any of these stages can

propagate into unsafe control actions or suboptimal operational decisions in safety-critical process environments.

However, unlike traditional model-based or statistical tools, LLMs are susceptible to hallucinations: fluent yet unsupported or incorrect explanations, diagnoses, or control recommendations. In safety-critical process control environments, such hallucinations pose significant risks to operational reliability, safety, and regulatory compliance. Addressing hallucinations in LLM-enabled PSE is therefore essential not only for improving technical performance, but also for ensuring trustworthy human–AI collaboration, safe automation of cognitive tasks, and sustainable industrial operation [6].

Most research on hallucinations has focused on factual or semantic errors in natural language generation for knowledge-centric tasks. In contrast, hallucinations in automation differ fundamentally: they manifest as incorrect control logic, fabricated causal assumptions, or implausible action sequences issued to real equipment, where consequences extend beyond textual error into material risk. Existing hallucination-detection methods developed for summarization, QA, or fact-verification are not directly applicable to this domain. White-box approaches, which rely on internal model activations [7,8], lack portability to deployed industrial models. Gray-box indicators such as token probability or entropy [9] poorly correlate with physical safety—high-confidence text may still violate pressure limits, fail to respect interlock logic, or misinterpret sensor drift. Classical black-box techniques [10,11] detect factual deviation but do not reason about thermodynamics, control constraints, or equipment sequencing. Even heuristic ensemble techniques such as self-consistency or sampling consensus [12] struggle when hallucinated instructions remain logically fluent while being operationally wrong.

To address this gap, we propose a neuro-symbolic abductive framework for detecting hallucination-driven control errors. Rather than validating commands through factual lookup or statistical confidence, the framework treats each LLM-generated control action as a hypothesis about how the plant should evolve. Counter-abduction is then used to generate competing hypotheses—alternative control sequences, explanations of observed sensor values, or safety-compliant causal interpretations. A command is considered hallucinated when it fails to survive this challenge: when a rival hypothesis aligns more closely with physical constraints, interlock logic, energy and mass balances, or risk-minimizing operation.

By defining hallucination as defeat under counter-abductive reasoning, this method shifts the detection problem from lexical correctness to causal soundness and physical validity. It operates entirely at the black-box level—neither model internals nor curated fact databases are required—yet it uncovers subtle control-logic hallucinations that remain undetected by probability-based or factual-verification approaches [13]. Importantly, this positions abduction not merely as a language-interpretation mechanism, but as a defensive layer protecting automation systems from confidently incorrect AI-generated actions.

1.1. Example: Hallucination ≈ Abductive Failure + Counter-Abductive Defeat

To illustrate how hallucinations manifest in automation systems, we analyze a concrete failure case in which an LLM generates a fluent but unsafe control recommendation. The example highlights how abductive reasoning and counter-abductive challenge serve as a basis for hallucination diagnosis within industrial environments.

Consider a continuous stirred-tank reactor (CSTR) operating at 92 °C, above its recommended safety threshold of 85 °C. The system queries an LLM-based supervisory agent: "Recommend control actions to reduce reactor temperature to safe operating limits."

The model responds with the following sequence (H_0—Hallucinated control plan):

1. Increase feed pump rate to dilute the reaction mixture.
2. Reduce agitation speed to minimize heat generation.
3. Maintain coolant flow constant for system stability.

The recommendation is linguistically coherent and uses appropriate process-control terminology. However, several actions contradict basic thermodynamic reasoning. Increasing feed rate can introduce more reactive material and exacerbate heat release; reducing agitation impairs heat transfer to the cooling jacket; maintaining coolant flow unchanged ignores the most direct lever for temperature reduction. Thus, although H_0 is plausible in form, it is physically unsound in function.

Abductive formulation acts as follow. Abduction asks whether a hypothesis exists under which the model's output would represent a coherent explanation of how cooling might be achieved. A repair hypothesis can be formulated as:

H_0 ′. *Increased feed acts as thermal dilution, and reduced agitation slows reaction rate sufficiently to offset heat accumulation. We then test H_0′ against established process knowledge (Table 1).*

Table 1. Evaluation criteria.

Evaluation Criterion	Result
Dilution decreases temperature faster than volume expansion increases heat load	Typically false for exothermic kinetics
Reduced agitation improves safety	Opposes heat-transfer principles; likely to worsen response
Constant coolant flow is sufficient	Violates standard safe-operation strategy

H_0′ therefore fails to explain the evidence without contradictions.

Counter-abductive defeat and hallucination recognition works as follows. Counter-abduction generates an alternative control hypothesis H^* grounded in mass and energy balance:

H^*—Physically valid response:

1. Increase coolant flow or enhance heat-transfer coefficient.
2. Reduce or suspend reactant feed.
3. Maintain or increase agitation to improve heat removal.

H^* aligns with core chemical-engineering principles and operational safety logic. Since H* explains system behavior more coherently and without contradiction—while H_0 cannot be reconciled with known thermodynamics—H_0 is defeated under counter-abductive challenge.

A process-automation hallucination is therefore defined as *a candidate control hypothesis that fails to survive counter-abductive evaluation against physically grounded alternatives*.

This mechanism establishes hallucination not as a linguistic artifact, but as a breakdown in causal reasoning and constraint-consistent action synthesis, enabling detection at the level of process logic rather than surface text.

1.2. Constrained Control

Constrained control is a cornerstone of industrial process control, reflecting the necessity to operate systems within explicit physical, safety, and operational limits such as actuator saturation, pressure and temperature bounds, and regulatory constraints. Among constrained control strategies, Model Predictive Control (MPC) has become the dominant industrial paradigm because it formulates control actions as a constrained optimization problem over a prediction horizon, explicitly enforcing input and state constraints while optimizing performance objectives [14]. Complementary tactics—including anti-windup

mechanisms for PID controllers, reference governors for constraint-safe set-point adjustment, and soft-constraint formulations with penalty terms—are widely used to ensure robust and safe operation under disturbances, model mismatch, and uncertainty [15,16]. Together, these methods form a safety envelope that limits the consequences of imperfect models and noisy measurements.

As LLMs are increasingly introduced into process control workflows—for supervision, fault explanation, operator assistance, or high-level decision support—the role of constrained control becomes even more critical. LLMs can generate fluent control rationales or recommendations that are not grounded in process physics or constraint logic, a phenomenon commonly referred to as hallucination [17]. In safety-critical control settings, such hallucinations may suggest infeasible set-points, unsafe operating regions, or incorrect interpretations of constraint violations. Consequently, LLM-driven control assistance must remain subordinate to constraint-aware control layers such as MPC and reference governors, and should be coupled with verification mechanisms that check consistency against process models and enforced constraints before any operational influence [18]. Without such safeguards, hallucinations risk undermining both operational safety and operator trust in AI-assisted control systems.

1.3. Contribution

To the best of our knowledge, this is the first work to systematically demonstrate how abductive reasoning can be used to detect and correct hallucinated control actions generated by LLMs in industrial process automation. We show that abduction provides a principled mechanism for constructing and testing operational hypotheses, enabling the system to recognize when an LLM's recommended actions rely on unsupported causal assumptions, violate physical constraints, or contradict established process logic. By extending this with counter-abduction—and grounding evaluation in thermodynamic, kinetic, and interlock-aware reasoning—we expose failure modes that are invisible to probability-based confidence scores, retrieval augmentation, or ensemble sampling methods. This positions abduction as a generalizable safety layer for LLM-driven automation, improving factual grounding, operational reliability, and the interpretability of control decisions in real industrial environments.

2. Abduction vs. Deduction and Induction

Charles S. Peirce [19,20] identified three fundamental modes of inference—deduction, induction, and abduction—that together constitute the foundation of human reasoning and scientific discovery. These reasoning forms differ in their epistemic orientation, logical direction, and ampliative strength (adds new information not contained in the premises). Peirce emphasized that deduction is non-ampliative and certain, while both induction and abduction are ampliative and uncertain, expanding knowledge beyond what is explicitly contained in the premises.

Deduction operates by applying general rules to specific cases to derive necessary conclusions. It preserves truth but does not introduce new information: if the premises are true, the conclusion must also be true. Deductive reasoning thus provides logical certainty but no epistemic novelty.

Induction, by contrast, generalizes from specific instances to formulate general laws or statistical regularities. It infers generality from repetition: from observing many cases of a phenomenon, one concludes that it holds universally or probabilistically. Inductive inference is inherently uncertain and probabilistic—its strength depends on the representativeness and size of the observed sample.

Abduction—sometimes called inference to the best explanation—reverses this direction of reasoning [21]. It begins with an observed or surprising fact and seeks the most plausible hypothesis that could account for it. Abduction is therefore explanatory and hypothetical in nature: it introduces a new conceptual assumption that, if true, would make the observed fact intelligible.

Peirce maintained that induction and abduction are irreducible to one another, despite both being ampliative. Their irreducibility lies in their distinct epistemic roles within inquiry. Abduction functions as the logic of discovery, generating explanatory hypotheses to make sense of observations. Induction functions as the logic of verification, evaluating the empirical adequacy of those hypotheses through repeated testing. Deduction mediates between the two, deriving predictions from the abductively generated hypotheses for inductive testing.

This cyclical relationship—*abduction* → *deduction* → *induction*—constitutes what Peirce regarded as the *complete logic of scientific inquiry*. In modern contexts such as abductive logic programming, probabilistic reasoning, and neuro-symbolic AI, this triad continues to serve as a conceptual framework for balancing hypothesis generation, logical inference, and empirical validation (Table 2).

Table 2. Forms of reasoning.

Form of Reasoning	Direction of Inference	Goal/Function	Type of Conclusion	Certainty	Example
Deduction	General → Particular	Apply rule to case	Necessarily true if premises are true	Certain	All metals expand when heated → Iron expands when heated
Induction	Particular → General	Formulate general law from observations	Probably true, subject to testing	Probabilistic	Iron, copper, and aluminum expand → All metals expand
Abduction	Particular → Hypothesis	Propose best explanation for observed fact	Plausible, explanatory conjecture	Uncertain	The surface of a reactor is wet → there was a leak and the humidity is high

3. Abduction as a Structural Corrective Layer for Reasoning About Processes

Chain-of-Thought (CoT) prompting has rapidly emerged as a default approach for eliciting multi-step operational reasoning from large language models. In process automation, CoT is often used to make the model articulate the intermediate control logic behind a recommended sequence of actions—e.g., why it suggests adjusting feed rate, coolant flow, or agitation in a particular order. However, empirical evidence shows that CoT frequently produces *narrative reasoning* rather than *causal reasoning*. Because CoT unfolds autoregressively, each step is conditioned primarily on previous text rather than on physical laws, kinetic models, interlock logic, or plant constraints. This leads to characteristic failure modes in automation: invented process states, circular justification of control actions, step sequences that violate safe operating procedures, and high variability when the same problem is phrased differently. The result is a reasoning chain that may appear methodical and domain-aware, yet lacks grounding in thermodynamics, mass transfer, or safety envelopes.

Abductive reasoning offers a structural countermeasure to these weaknesses. Unlike deduction, which derives consequences from known rules, or induction, which extrapolates from prior experience, abduction seeks the *most plausible causal hypothesis* for a given set of process observations—sensor readings, alarm states, rate deviations, or temperature excursions. Integrated with LLMs, abduction provides a corrective layer that forces generative control rationale to align with physical constraints and interlock logic. Rather than merely checking CoT after generation, abductive integration reshapes the reasoning trajectory itself, yielding multi-step control explanations that are coherent, defeasible, and grounded in process reality.

In a neuro-symbolic pipeline, the role of abduction is to constrain the model's reasoning space, reveal implicit assumptions, and ensure that the chain as a whole satisfies the explanatory minimality principles characteristic of abductive logic programming and related frameworks (e.g., probabilistic logic programming, argumentation-based abduction, and paraconsistent abduction). The resulting system treats CoT not as a static artifact but as a dynamic structure subject to revision, hypothesis insertion, and consistency checking. This greatly mitigates classical CoT hallucinations, particularly those involving unjustified intermediate premises.

LLMs demonstrate several recurrent weaknesses when generating multi-step control logic for industrial processes:

1. Local coherence without global constraint compliance. Autoregressive reasoning produces steps that sound plausible in isolation, yet the overall control plan may violate material balances, safety limits, or standard operating procedures, leading to hidden contradictions across the sequence:
2. Operational drift. A control recommendation may begin sensibly—e.g., increasing coolant flow—but gradually diverge into irrelevant or unsafe actions (changing feed composition, reducing agitation, bypassing interlocks) as the reasoning chain extends.
3. Fabricated plant-state assumptions. To maintain fluency, models often insert implicit premises—unverified sensor readings, nonexistent equipment states, or speculative causal relationships—not grounded in actual process data.
4. Irreversibility of reasoning. Chain-of-thought control plans are monotonic: once a step is generated (e.g., reduce feed), the model rarely retracts or corrects it even when subsequent reasoning contradicts that action or new evidence invalidates it.
5. Non-minimal, overly verbose action paths. Generated control sequences frequently include redundant or unnecessary steps, increasing execution risk and inflating the opportunity space for hallucination and unsafe actuation.

These failure patterns stem from the lack of symbolic, constraint-aware structure in LLM-generated control logic—the core instance of the language-model fallacy, where linguistic plausibility is mistaken for physical validity. Abductive reasoning directly targets these failure modes, introducing a mechanism to test, revise, and compete control hypotheses against thermodynamic, interlock, and mass-balance realities.

Abductive CoT

Abductive models enforce minimality: explanations should contain no unnecessary assumptions. This principle acts as a structural regularizer on CoT, pruning verbose or extraneous content and discouraging speculative detours. Minimality also makes verification more tractable because the reasoning chain becomes closer to a canonical explanation.

Moreover, minimality reduces one of the main sources of hallucination in CoT systems: the inclusion of tangential premises or loosely associated facts. A minimal abductive explanation is not only easier to inspect but also more robust to adversarial perturbations and paraphrased prompts.

A coherent architecture for Abductive CoT emerges from combining these elements (Figure 2):

1. Initial CoT generation by the LLM.
2. Logical extraction converting text into predicates or defeasible rules.
3. Abductive solver evaluates consistency, minimality, and coherence.
4. Hypothesis generation to fill explanatory gaps.
5. Feedback to LLM prompting revision or alternative reasoning paths.
6. Discourse-aware weighting using RST to distinguish central from peripheral content.
7. Final, verified CoT chain that satisfies explanatory constraints.

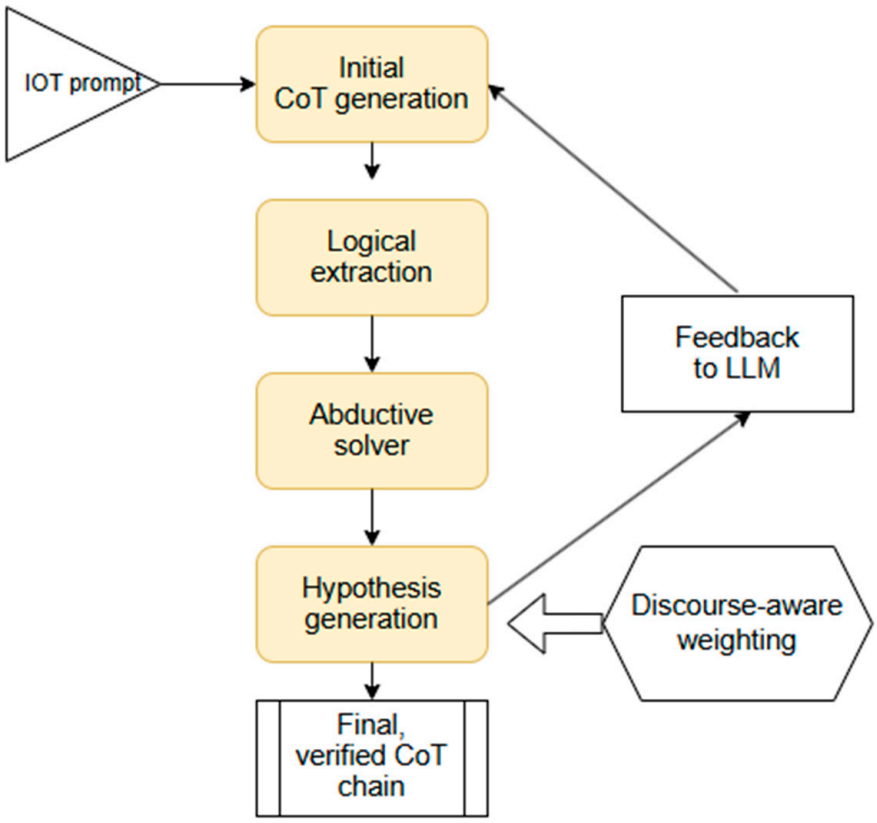


Figure 2. Abductive support for CoT.

This loop is compatible with multiple logical formalisms, including probabilistic abduction, argumentation-based abduction, and paraconsistent abductive reasoning—allowing different degrees of uncertainty, conflict tolerance, and rule expressiveness. The core advantage is that the LLM no longer bears the full burden of reasoning; instead, it operates within a scaffold of symbolic constraints.

4. Abductive Logic Programming

In Abductive logic programming (ALP), one allows some predicates (called *abducibles*) to be "hypothesized" so as to explain observations or to achieve goals, subject to integrity constraints. An abductive explanation is a set of ground abducible facts Δ such that:

1. $P \cup \Delta \models G$ (i.e., the goal/observation G is entailed);
2. $P \cup \Delta \models IC$ (the integrity constraints are satisfied);
3. $P \cup \Delta P$ is consistent.

Here $<P,A,IC>$ is the abductive logic program: P is the normal logic program, A the set of abducible predicates, and IC the constraints.

There are many ALP systems available (Table 3).

Table 3. Abductive logic programming systems.

Name	Approach/Features	Notes/Strengths	Limitations/Caveats
ACLP	Integrates abduction with constraint solving (built over ECLiPSe CLP)	Good fit when you need both abduction and constraints (e.g., planning, scheduling).	Performance can degrade for large or complex abductive tasks.
CIFF/IFF-based systems	Use a variant of the IFF proof procedure extended with abductive reasoning and constraints	More expressive handling of integrity constraints, etc., widely referenced in ALP literature	As with many meta-interpreters, efficiency is a concern for large domains.
A-system	A Prolog-based abductive system	One of the classical ALP systems.	Might not scale to very large problems; also dependent on the Prolog engine.
SCIFF	An extension of ALP tailored for specifying and checking protocols (e.g., interaction, contracts)	Good for normative reasoning, protocol compliance monitoring.	Specialized; might require tailoring for more general domains.
ABDUAL	A system combining abduction and tabling techniques [22]	Helps in improving efficiency, avoiding redundant recomputation.	Implementation complexity; tradeoffs in memory vs. speed.
DLV (with abductive diagnosis front-end)	DLV is a disjunctive ASP/nonmonotonic reasoning system; it supports a front end for abductive diagnosis tasks.	Leverages efficient ASP back ends; good for problems reducible to abductive diagnosis.	May require rephrasing of your problem into the dialect ASP supports; constraints of DLV's language.
ToyElim	A more general system for operator elimination (e.g., quantifier elimination, projection, and forgetting) which can express abductive explanations.	Elegant, theoretically grounded in classical logic; may serve as a backend or bridge.	It is a prototype; may not be optimized for large logic programming tasks.

There are Prolog-based approaches/tools that support or partially support abductive reasoning/ALP. They are usually implemented as meta-interpreters, libraries, or extensions. We mention three families of approaches:

- Aleph (with "abduce" mode). Aleph is primarily an Inductive Logic Programming (ILP) system. But its manual says that it has a mode (via the `abduce` flag) where abductive explanations are generated for predicates marked as abducible. The abductive part in Aleph is limited: it assumes abducible explanations must be *ground*, and you may need to limit the number of abducibles (via `max_abducibles`) for efficiency [23].
- Meta-interpreter/CHR implementations in Prolog. Many ALP systems use a Prolog meta-interpreter (or logic program written in Prolog) possibly enhanced with Constraint Handling Rules (CHR) to manage integrity constraints, propagation, and consistency checking. Since SWI-Prolog supports CHR (via its CHR library/attributed variables), you can port or build an abductive system using CHR in SWI [24].

It is possible to build a meta-interpreter for ALP directly. The general approach:

1. declare which predicates are abducibles,
2. write a meta-interpreter that, when trying to prove a goal, allows adding abducible atoms hypotheses,
3. maintain integrity constraints and check them,
4. control search (pruning, minimality, consistency).

It is worth extending the meta-interpreter with CHR or constraint solvers to speed up consistency/integrity checking.

Some recent proposals aim to make ALP systems more efficient (e.g., by eliminating CHR overhead) or compile them, but they may not yet have full, robust SWI-Prolog ports. Also, SWI Prolog has features like attributed variables, constraint libraries, and delimited control (in newer versions) which facilitates more advanced meta-programming approaches useful in ALP. Several methodological and computational challenges are associated with the use of ALP:

1. Scalability remains a central issue. Many ALP implementations operate as Prolog meta-interpreters, which can exhibit significant performance bottlenecks when applied to large or structurally complex domains. Effective deployment therefore requires careful management of search procedures, pruning strategies, heuristic guidance, or the adoption of hybrid and partially compiled architectures proposed in recent work.
2. Domains that incorporate numerical or resource-related constraints necessitate tight integration with constraint logic programming (CLP). Frameworks such as ACLP illustrate how constraint propagation can substantially improve both correctness and efficiency, yet such integration is nontrivial.
3. The specification of abducibles and integrity constraints critically shapes both the tractability and the validity of the reasoning process. Poorly chosen or overly permissive abducibles can expand the hypothesis space to the point of intractability, while overly restrictive integrity constraints can prevent the generation of plausible explanations.
4. Although many abductive tasks can be reformulated as Answer Set Programming (ASP) problems and thus leverage highly optimized ASP solvers, doing so typically requires nontrivial representational transformations [25]. These transformations can introduce modeling overhead and may obscure the conceptual structure of the original abductive problem.

Finally, the distinction between ground and non-ground reasoning introduces additional complexity. Systems optimized for propositional, fully grounded settings often achieve superior performance, whereas support for variables, unification, and non-ground abductive hypotheses tends to complicate search and reduce scalability. Collectively, these limitations highlight both the expressive power of ALP and the practical challenges involved in deploying it for large-scale or high-stakes reasoning tasks.

The computational pipeline for Abductive Logic Programming (ALP), illustrated in Figure 3, proceeds through the following stages.

1. Discourse Parsing. The input explanation text is first segmented into Elementary Discourse Units (EDUs) and organized into a discourse tree capturing nucleus–satellite relations and rhetorical roles. For high-fidelity, low-volume analyses we employ GPT-5 for discourse parsing, which provides robust identification of causal, evidential, and contrastive relations. For large-scale evaluation, we use a lightweight wrapper around the discourse parser of [26], trading some granularity for scalability while preserving core rhetorical structures.

2. Fact Extraction. Each EDU is then mapped into one or more logical literals representing causal relations, evidential support, contextual qualifiers, or contrastive constraints. This step grounds natural-language explanations in a symbolic representation suitable for reasoning, ensuring that all inferred relations are explicitly traceable.
3. Weight Assignment. Discourse roles are converted into numerical weights, with nucleus units receiving higher scores than satellites. These weights reflect the relative explanatory importance of different discourse elements, prioritizing causal and evidential statements over background or elaborative content, while allowing contrastive units to introduce defeaters.
4. Abductive Search. The weighted literals are passed to an abductive reasoning engine implemented using SWI-Prolog and a probabilistic ALP framework (e.g., ProbLog). The solver generates candidate hypotheses that explain the observed facts while minimizing abductive cost and respecting discourse-weighted constraints.
5. Ranking and Selection. Finally, candidate explanations are ranked by their aggregated weighted scores, and the top-k hypotheses are returned. These ranked hypotheses provide both the most plausible explanations and an explicit ordering of alternatives, enabling counter-abductive analysis and hallucination detection.

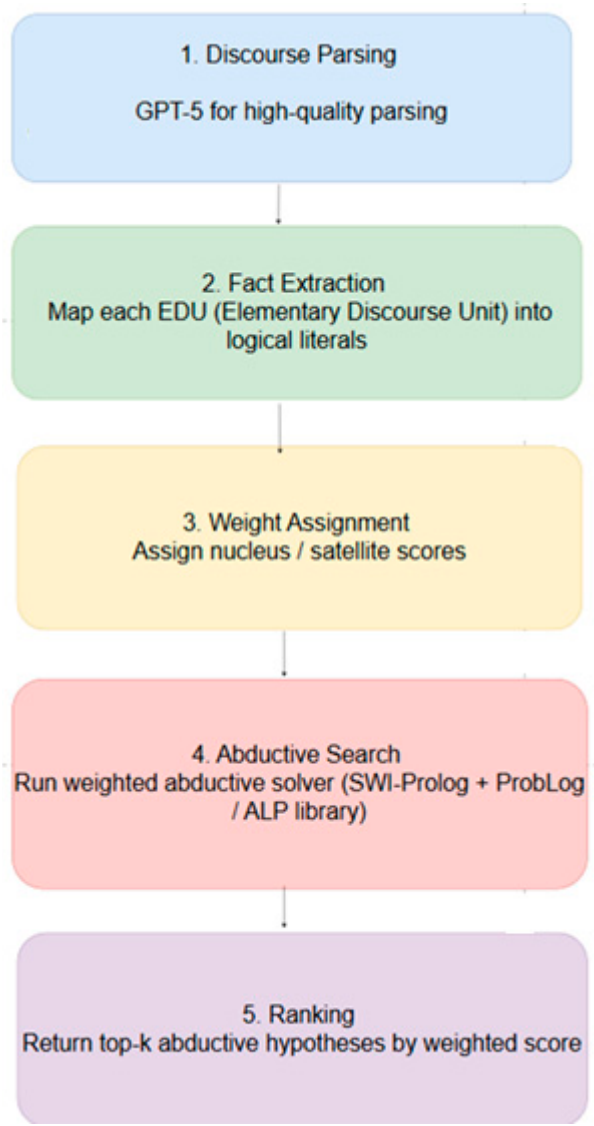


Figure 3. Computational pipeline.

5. Counter-Abduction

The role of counter-abduction in neuro-symbolic reasoning is best understood by tracing its origins to classical accounts of abductive inference and modern theories of confirmation. Abduction denotes the inferential move in which a reasoner proposes a hypothesis H that, if true, would render a surprising observation E intelligible. Peirce emphasized that abduction is neither deductively valid nor inductively warranted; its justification lies in explanatory plausibility rather than certainty.

However, in both human and machine reasoning, the first abductive hypothesis is often not the most reliable. This motivates the introduction of *counter-abduction*, a concept developed implicitly in sociological methodology and more formally in abductive logic programming [27]. Counter-abduction refers to the generation of alternative hypotheses that likewise explain the evidence, thereby challenging the primacy of the initial explanation. For example, while an explosion may abductively explain a loud bang and visible smoke, counter-abductive alternatives—such as a car backfire combined with smoke from a barbecue—demonstrate that multiple explanations can account for the same phenomena [28].

To evaluate these competing hypotheses, the framework draws on *confirmation theory*, which provides probabilistic and logical tools for assessing evidential support [29]. In Bayesian terms, evidence E confirms hypothesis H if it increases its probability, i.e., if $P(H \mid E) > P(H)$. Probability-increase measures such as $d(H,E) = P(H \mid E) - P(H)$ and ratio-based measures such as $r(H,E) = P(H \mid E)/P(H)$ quantify the extent of confirmation [30]. Likelihood-based measures, including the likelihood ratio $P(E \mid H)/P(E \mid \neg H)$, further assess how much more expected the evidence is under the hypothesis than under alternatives. These tools allow structured comparison of hypotheses $\{H_1, H_2, \ldots\}$ generated via abduction and counter-abduction.

Chemical Process Example

Consider a continuous stirred-tank reactor (CSTR) exhibiting an unexpected temperature rise accompanied by reduced conversion efficiency. An LLM-based diagnostic component might abductively propose the hypothesis H_1: "the temperature controller is mis-tuned, causing oscillatory overheating." While H_1 plausibly explains the observed temperature deviation, counter-abductive reasoning generates alternative hypotheses, such as H_2: "an unmeasured increase in reactant feed concentration has elevated the reaction rate," or H_3: "fouling in the heat exchanger has reduced heat removal capacity." All three hypotheses can account for the same evidence E (temperature increase and efficiency loss), yet they imply fundamentally different corrective actions.

Using counter-abduction, the framework evaluates which hypothesis is best supported by additional process evidence and integrity constraints. For example, stable controller outputs may weaken H_1, while unchanged feed analyzer readings may partially disconfirm H_2. Meanwhile, a gradual increase in coolant-side pressure drop provides confirmatory evidence for H_3. Likelihood ratios and probability updates allow the system to rank these explanations, exposing cases where the initial abductive hypothesis—although fluent and operationally plausible—lacks sufficient evidential support. In the context of hallucination detection, this mechanism is critical: an LLM-generated explanation may satisfy process constraints and appear coherent, yet be rejected through counter-abduction because a rival hypothesis better explains the observed data.

This example illustrates how counter-abduction functions as a safeguard against premature or overconfident explanations in process control settings. By explicitly generating and comparing competing causal accounts, the framework prevents single-hypothesis dominance and enables principled rejection of unsupported narratives. When integrated with LLM-

assisted diagnostics, counter-abduction thus plays a central role in detecting explanation-level hallucinations, ensuring that accepted interpretations remain grounded in process physics, measurements, and causal structure rather than linguistic plausibility alone.

6. Discourse in Abductive Logic Programming

ALP is designed to generate hypotheses (abducibles) that, when added to a knowledge base, explain observations. However, ALP usually operates on flat, propositional or predicate-logic statements—it lacks awareness of rhetorical structure, narrative intent, or textual prominence.

Discourse analysis, especially based on RST, gives us a hierarchy of rhetorical relations between text segments—e.g., *Cause–Effect*, *Condition*, *Evidence*, *Contrast*, *Elaboration*. Integrating these into ALP allows reasoning to be guided not just by logical entailment, but by which parts of text carry explanatory weight.

Conceptual integration is shown in Table 4.

Table 4. Extending the features of ALP with discourse information.

Aspect	In Classical ALP	With Discourse-Aware ALP
Observation	A set of atomic facts or predicates.	Clauses extracted from *nucleus* discourse segments (main claims).
Abducibles	Candidate explanatory literals.	Hypotheses aligned with *satellite* segments, weighted by rhetorical relation (e.g., Evidence ↑, Elaboration ↓).
Explanatory Preference	Minimality or cost-based.	Weighted abductive preference: prioritize hypotheses supported by nucleus–satellite strength and coherence relations.
Conflict Resolution	Based on logical consistency.	Also guided by discourse coherence: conflicting explanations that preserve discourse flow are preferred.

We now proceed to the example: “The unit triggers a high-temperature alarm at the outlet and a low-flow alarm on the coolant line. Since the deviation began right after a post-maintenance startup and the coolant flow is unstable, a pump failure is likely.”

Discourse parsing output: Nucleus (primary symptoms/alarms):

“The outlet high-temperature alarm is active and the coolant low-flow alarm is active.”

Satellite (Cause–Effect/context):

“Since the event began after post-maintenance startup and the coolant flow shows instability.”

Claim (Evaluation):

“Pump failure is likely.”

ALP representation is as follows:

```
% --- Background Knowledge (Process Control Causal Relationships) ---

cause(high_feed_temp, increased_outlet_temp).
cause(low_coolant_flow, increased_outlet_temp).
cause(fouling, reduced_heat_transfer).
cause(reduced_heat_transfer, increased_outlet_temp).

cause(pump_failure, low_coolant_flow).
cause(valve_stiction, fluctuating_flow_rate).

symptom(increased_outlet_temp, high_temp_alarm).
symptom(low_coolant_flow, low_flow_alarm).
symptom(fluctuating_flow_rate, oscillatory_control_signal).

% Useful intermediate rules
triggers(X, Y) :- cause(X, Y).
leads_to(X, Y) :- cause(X, Y).
indicates(Observation, Fault) :- symptom(Fault, Observation).

% --- Observations from Process Historian / SCADA ---
obs(high_temp_alarm).
obs(low_flow_alarm).
obs(post_maintenance_startup).     % relevant contextual fact
obs(flow_rate_instability).
% Discourse weighting
nucleus_weight(1.0).
satellite_weight(0.6).
% Abductive rule (discourse-aware)
explain(Obs, Hyp) :-
    nucleus(Obs, Nuc), satellite(Obs, Sat),
    abduct(Hyp),
    satisfies(Nuc, Hyp, W1),
    satisfies(Sat, Hyp, W2),
    Score is W1*1.0 + W2*0.6,
    Score > Threshold.
```

Here the *nucleus* (joint pain, swelling) gives hard constraints, while the *satellite* (seafood meal cause) provides softer evidence with lower weight [31]. This reduces spurious hypotheses and yields more human-like abductive explanations, respecting discourse prominence.

6.1. Discourse-Weighted ALP (D-ALP)

Let $P = (\Pi, \Delta, A)$ be a standard abductive logic program:

- Π—strict rules;
- Δ—defeasible rules;
- A—set of abducibles.

We extend it with a discourse weighting function w: $L \in [0, 1]$ over literals L derived from RST trees:

- $w(l) = 1.0$ if l originates from a nucleus clause;
- $0 < w(l) < 1$ if l originates from a satellite clause;
- $w(l) = 0$ if l appears in background or elaborative relations.

Then the abductive explanation $E \subseteq A$ minimizes:

$$\mathrm{Cos}(E) \;=\; \sum_{l \in \mathrm{E}} (1 - w(l)) \cdot penalty(l)$$

subject to $\Pi \cup E \models O$.

Thus discourse prominence directly affects the search space and preference ordering among explanations.

The penalty function *penalty*(l) quantifies how "expensive" it is to *abduce* literal l—i.e., to assume l as true when it is not derivable from the strict rules Π. It represents epistemic risk: how far l departs from evidence, domain priors, or discourse plausibility.

$$\mathrm{penalty}(l) = \alpha \cdot \rho(l) + \beta \cdot \kappa(l) + \gamma \cdot \delta(l),$$

where

- $\rho(l)$ is a rule distance—number of rule applications needed to derive l (depth in derivation tree).
- $\kappa(l)$ is a conflict measure—degree to which l contradicts existing facts or competing hypotheses.
- $\delta(l)$ is a discourse mismatch measure—how incompatible l is with its rhetorical context (e.g., "Contrast", "Condition").
- Constants α, β, γ control the importance of logical vs. discourse penalties (often $\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$).

Thus, penalty(l) is higher for:

- hypotheses that are logically remote;
- contradict evidence;
- misalign with the discourse flow.

Assume a process event log produces three competing fault hypotheses: coolant-flow restriction, heat-exchanger fouling, and sensor drift. Each is associated with different abductive characteristics. The system identifies coolant-flow restriction as the primary (nucleus) hypothesis: it has low rule distance from observations ($\rho \approx 0.1$), no direct contradictory evidence ($\kappa \approx 0$), and minimal discourse mismatch with the operator narrative ($\delta \approx 0.1$), yielding a small normalized penalty of roughly 0.12.

Heat-exchanger fouling appears as a competing hypothesis: although similarly plausible in principle ($\rho \approx 0.1$), it carries observable conflict—recent maintenance contradicted the expectation of fouling—and a moderate discourse mismatch ($\kappa \approx 0.4$, $\delta \approx 0.2$). This raises the penalty to approximately 0.22.

The third hypothesis, sensor drift, functions as an irrelevant satellite explanation. It is weakly connected to the alarm pattern and has greater inconsistency with recorded temperature and flow signals ($\rho \approx 0.3$, $\kappa \approx 0.5$, $\delta \approx 0.7$). Its penalty rises to nearly 0.43, making it the least supported explanation of the three.

Computing discourse-weighted abductive cost (D-ALP) produces the following pattern of preference:

- If w(*coolant-flow-restriction*) = 1.0, the cost approaches 0;
- If w(*fouling*) = 0.7, the cost is approximately 0.066;
- If w(*sensor-drift*) = 0.4, the cost increases to ≈0.26.

Thus, D-ALP selects coolant-flow restriction as the most plausible root cause—lowest penalty, strongest discourse alignment, and highest diagnostic confidence.

6.2. Combining Discourse Analysis with Explainable Reasoning

We combine discourse analysis with explainable reasoning frameworks such as ALP, argumentation, or probabilistic logic. We propose how the structure of a discourse tree can

be used to extract, classify, and weight different forms of explanations from text or model outputs. When a discourse tree with EDUs and relations is mapped into logical structures, each subtree becomes a candidate explanatory unit—either *causal*, *evidential*, *conditional*, or *contrastive*. Thus, the discourse tree serves as a blueprint for structured explanation extraction. The mapping is shown in Table 5.

Table 5. From discourse to explanation types.

Discourse Relation	Typical Explanation Type	Logical Template
Cause/Result	Causal explanation	$E \leftarrow cause(X,Y)$
Evidence/Justify	Evidential explanation	$E \leftarrow supports(X,Y)$
Condition/Hypothetical	Conditional explanation	$E \leftarrow if(X)\ then(Y)$
Contrast/Antithesis	Counterfactual or contrastive explanation	$E \leftarrow prefer(X,Y)$
Elaboration/Background	Contextual explanation	$E \leftarrow context(X,Y)$
Purpose/Motivation	Goal-oriented explanation	$E \leftarrow intent(X,Y)$

In this framework, discourse relations are not treated as merely rhetorical structures but are assigned explicit weights reflecting their diagnostic importance in explanation verification. Relations linked to the nucleus of the discourse—such as *Cause/Result* and *Evidence/Justify*—receive the highest weights, as they directly support the core explanatory claim and correspond to physically grounded causal or evidential links in the process. Satellite relations, including *Elaboration/Background* and *Purpose/Motivation*, are weighted lower because they provide contextual or operational background rather than direct causal support. *Condition/Hypothetical* relations receive intermediate weights, reflecting their role in qualifying explanations without asserting actual causation. *Contrast/Antithesis* relations are treated asymmetrically: while they do not increase support for a hypothesis, they contribute negatively by enabling counter-abductive defeat when a competing explanation better accounts for the observed data. By weighting nucleus and satellite relations differently, the system prioritizes explanations that are causally and evidentially grounded while remaining sensitive to contextual qualifiers and explicit counter-evidence, thereby improving robustness against superficially plausible but incorrect explanations.

Given a discourse tree with nodes—EDUs, the following steps are performed:

1. Parse the text → obtain discourse tree.
2. Identify nucleus/satellite pairs → mark *nucleus* as potential hypothesis/conclusion, *satellite* as premise/explanation.
3. Label each edge with its rhetorical relation.
4. Translate edges to logical relations, as follows.

```
cause(X, Y). % from Cause relation
evidence(X, Y). % from Evidence relation
condition(X, Y).
contrast(X, Y).
```

Subtrees are aggregated according to the following rules:

- A *deep* subtree (multiple Evidence nodes) → composite explanation;
- A *flat* tree (multiple Causes) → multi-factor explanation;
- A *branch with Contrast* → comparative explanation.

Consider the following diagnostic narrative: "The reactor outlet temperature has risen sharply. Since this increase followed a drop in coolant flow, and heat-transfer effi-

ciency is low, pump restriction is suspected. However, vibration readings do not indicate impeller damage."

Discourse Tree (simplified):

```
          [Pump restriction suspected]
                    /            \
          (Evidence)             (Contrast)
          /        \                  \
 [Low coolant]  [Low heat-transfer]   [No vibration anomaly]]
```

From this, we extract:

```
explain(pump_restriction,cause(low_coolant_flow, increased_temperature)).
explain(pump_restriction, evidence(low_heat_transfer_efficiency)).
explain_not(pump_restriction, contrast(no_vibration_detected)).
```

These can be ranked or combined into a structured abductive hypothesis:

E ={cause(A,B),evidence(C),¬contrast(D)}, which forms a multi-faceted explanation for the same claim.

Different *forms of explanation* (causal, evidential, contextual, counterfactual) can be systematically extracted by pattern-matching substructures of the discourse tree (Table 6).

Table 6. The shape and depth of the discourse tree are important for ALP construction.

Structural Feature	Explanation Property	Interpretation
Depth (levels below nucleus)	Explanation completeness	deeper = more elaborated reasoning chain
Branching factor	Explanation diversity	multiple satellites = multi-evidence explanation
Contrastive nodes count	Explanation contestability	High → balanced/counterfactual reasoning
Causal path length	Explanation causal depth	long paths correspond to mechanistic explanations

When an ALP is formed:

1. Each explanation type defines a different abductive schema:
   - *Causal explanations* suggest new rules (A → B).
   - *Evidential explanations* add support to hypotheses.
   - *Contrastive explanations* add defeaters or alternative hypotheses.
2. The discourse tree structure provides hierarchical weighting for hypothesis generation.
3. The abductive search space becomes tree-constrained rather than flat—drastically improving interpretability and efficiency.

   By leveraging discourse trees, we can automatically construct:
   - Level 1 (Direct): nucleus-satellite explanation ("*Acidic PH because of uric acid* ↑")
   - Level 2 (Intermediate): composite explanation combining several satellites
   - Level 3 (Meta): contrastive or counterfactual explanation derived from opposing branches
   - Level 4 (Global): entire discourse tree summarized as an explanatory graph, suitable for argumentation or visualization.

7. Counter-Abduction Hallucination Detection Algorithm

While abduction has been widely studied as an explanatory mechanism, the complementary concept of *counter-abduction*—the generation and evaluation of rival hypotheses against an initial abductive explanation—has received less attention in the context of LLM reasoning.

In a neuro-symbolic setting, counter-abduction plays a crucial role in hallucination detection. It introduces an adversarial explanatory pressure on the LLM-generated CoT,

challenging its assumptions, testing its logical validity, and evaluating whether alternative explanations better satisfy coherence and minimality conditions. This section formalizes the relationship between counter-abduction and hallucination, illustrates how counter-abductive reasoning exposes invented premises and inconsistencies, and presents a unified algorithm for counter-abductive hallucination detection within the Abductive Chain-of-Thought (A-CoT) framework.

The algorithm takes as input an LLM-generated CoT, a symbolic knowledge base KB, a discourse-structure representation RST, and a set of abductive/counter-abductive generators. It outputs (1) whether the CoT is hallucinated, (2) the defeat score, and (3) a revised reasoning chain if needed.

The algorithm for counter-abductive hallucination detection is as follows:

Input:

- CoT explanation E_0;
- Knowledge base KB;
- Discourse structure RST with weights $w(\cdot)$;
- Abductive hypothesis generator A;
- Counter-abductive generator C;
- Logical verifiers $V = \{V_1, \ldots, V_n\}$.

Output:

- Hallucination label $H \in$ {True,False};
- Defeat score $D(E_0)$;
- Suggested corrected explanation E^*.

Procedure:

```
1. Parse E0 into atomic propositions P = extract(E0)

2. Identify surprising or inconsistent elements:
      S ← detect_surprises(P, KB, RST)

3. Generate abductive hypotheses for each surprise:
      H_A ← ∪{s ∈ S} A(s, KB)

4. Generate counter-abductive rival explanations:
      H_C ← ∪{s ∈ S} C(s, KB)

5. For each explanation h ∈ H_A ∪ H_C:
      - Evaluate consistency score:
            Cons(h) = consistency_check(h, KB)
      - Evaluate discourse alignment score:
            Disc(h) = Σ_{p ∈ h} w(RST(p))
      - Combine:
            Score(h) = α·Cons(h) + β·Disc(h)

6. Compute defeat of the original explanation:
      D(E0) = aggregate_{Vi ∈ V} Vi.compare(E0, H_C)

      where each Vi produces:
            Vi.compare(E0, H_C) =
                  1 if ∃ h ∈ H_C such that Score(h) > Score(E0)
                  0 otherwise

7. If D(E0) > τ (default τ = 0.5):
       H ← True            // hallucinated
       E* ← argmax_{h ∈ H_C} Score(h)
   Else:
       H ← False
       E* ← E0

8. Return (H, D(E0), E*)
```

The steps of counter abduction computation procedure are shown in Figure 4.

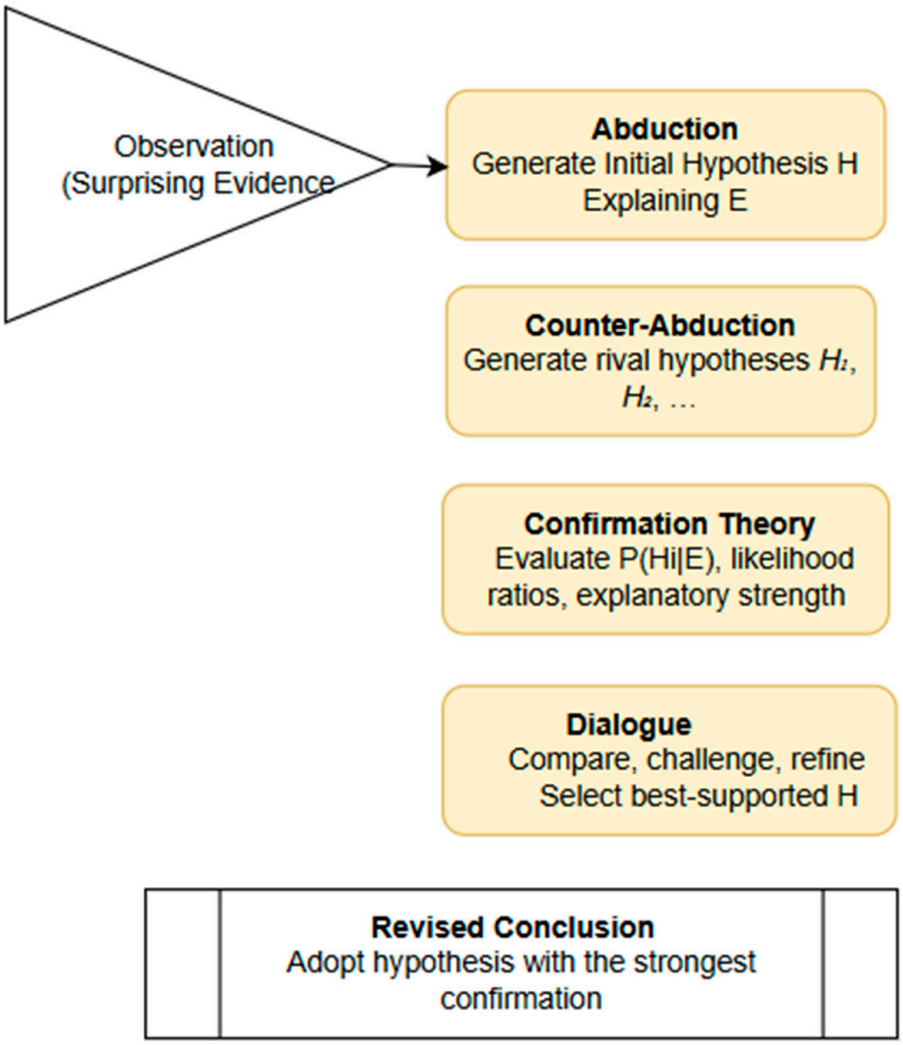


Figure 4. Steps of counter abduction.

8. Evaluation

We present a comprehensive evaluation of the proposed Discourse-Based Abductive Logic Programming (D-ALP) framework. The evaluation examines both computational performance and human-perception aspects. The computational evaluation measures the accuracy, efficiency, and consistency of the abductive reasoning process when discourse structure is integrated. The human evaluation investigates how users perceive clarity, coherence, and trust in discourse-aware explanations.

Our focus is hallucination detection for model answers using four logical assessment methods as validators. Each validator assesses whether an answer's central claim is defeated by the argument-validation system. All datasets were annotated with rhetorical roles such as nucleus and satellite, and rhetorical relations such as *Cause*, *Evidence*, *Contrast*, *Condition*, *and Elaboration*. These annotations guide D-ALP's discourse weighting.

We define a hallucination as a claim whose defeat probability exceeds 0.5. This cautious threshold is motivated by safety-critical domains, where we prefer to reject answers that are defeated with substantial probability.

8.1. Evaluation Dataset

To evaluate explanation-level hallucinations in industrial settings, we constructed a new Process-Control Hallucination Dataset consisting of 500 synthetic but realistic diagnostic scenarios derived from common faults in chemical and energy systems (e.g., pump restriction, valve stiction, heat exchanger fouling, compressor surge, sensor drift, and reactor runaway). Each scenario includes: (i) a structured set of process signals and contextual information (temperatures, flows, vibration, controller patterns, maintenance notes); (ii) the true underlying fault; (iii) a correct mechanistic explanation grounded in process physics; (iv) a plausible but incorrect hallucinated explanation that an LLM might generate; (v) an easy-but-wrong explanation, designed to appear superficially credible while contradicting operational evidence; and (vi) explicit counter-evidence supporting counter-abductive reasoning.

A sample record includes: Each item includes:

1. Context (signals + logs);
2. True fault;
3. Correct mechanistic explanation;
4. Hallucinated explanation (plausible but wrong);
5. Easy-but-wrong explanation (dangerous, highly seductive);
6. Counter-evidence (for counter-abduction).

The record example is below:

```
{
  "context": {
    "temp_out": 145,
    "flow_coolant": 40%,
    "vibration": normal,
    "pressure_drop": minimal,
    "maintenance_log": "cleaned heat exchanger two days ago"
  },
  "true_fault": "pump_restriction",
  "correct_explanation": "The outlet temperature rise follows coolant flow reduction, not fouling. No vibration anomaly supports pump impeller integrity.",
  "hallucinated_explanation": "Heat exchanger fouling is increasing thermal resistance."   // appears plausible but incorrect
}
```

The dataset (https://github.com/bgalitsky/halluc_in_health/blob/master/data/process_control_hallucination_500.json, accessed on 16 January 2026) mirrors real diagnostic practice in process control, where multiple fault hypotheses may appear consistent with partial observations, and where misleading yet intuitive narratives frequently arise. By encoding mechanistic, evidential, and contrastive structures, the dataset enables fine-grained evaluation of explanation fidelity—not just fault identification. This resource therefore provides a controlled testbed for comparing black-box, retrieval-based, deliberation-based, and neuro-symbolic systems. In particular, it allows us to probe whether a model can distinguish between valid causal chains and structurally incorrect but plausible explanations, a key challenge in high-stakes autonomous monitoring and decision support.

We observe the following hallucination types (Table 7).

The dataset is constructed by combining synthetic and, where available, real process-control data to create diagnostically realistic scenarios. Synthetic plant data are generated from simplified but physically grounded models such as two-stream heat exchangers and closed-loop cooling systems, using simulation environments including Modelica (OpenModelica: latest release 1.22.2), Aspen HYSYS (V. 15), or MATLAB Simulink (v. 24.2) to produce time-series signals, alarms, and contextual events. For each simulated fault or operating condition, we generate paired explanations: a correct mechanistic explanation consistent with process physics, a plausible but incorrect hallucinated explanation that fits surface patterns, and harder cases containing conflicting or partial evidence designed to trigger misleading inferences. Each scenario is then annotated using a discourse-aware abductive representation, explicitly encoding causal links, supporting evidence, and contrastive signals that weaken a hypothesis. Finally, scenarios are labeled by difficulty, ranging from single-cause faults to multi-causal and deceptive cases with strong hallucination triggers. In evaluation, an LLM is tasked with producing both a fault diagnosis and its underlying reasoning, which is then assessed for whether the explanation is grounded in the encoded causal structure or constitutes a hallucinated narrative.

Table 7. Hallucination types in our dataset.

Type	Example
Mechanistic hallucination	Inferring fouling from high ΔT without flow or pressure drop evidence
Temporal hallucination	Reversing cause-effect (ex. "temperature rise caused the flow reduction")
Symmetry hallucination	Assuming conditions upstream mirror downstream readings
Sensor substitution hallucination	Treating one sensor's reading as if it measured a different variable
Counterfactual blindness	Ignoring conflict evidence like stable vibration or normal pump power draw

8.2. Experimental Setup

Five systems were compared (Table 8). GPT 5.1 was used for evaluation.

Table 8. Approaches to hallucination detection evaluated in this study.

System	Description
Baseline ALP	Classical abductive logic programming with uniform rule weights
ProbALP	Probabilistic ALP without discourse cues
D-ALP-Depth	Depth-weighted abduction using nucleus/satellite hierarchy
D-ALP-Rel	Relation-aware abduction using rhetorical relation types
Full D-ALP	Combined hierarchical and relational weighting (proposed system)
Abductive RAG	Framework that integrates abductive inference into retrieval-augmented LLMs [32]

Evaluation metrics included logical accuracy (F1), average reasoning time, the proportion of defeated hypotheses (logical inconsistency), and human ratings of clarity, coherence, and trust. All numerical results were averaged across ten runs and three datasets.

8.3. Hallucination Detection

We evaluate on three claim-verification datasets that we derive from existing QA/NLI resources: TruthfulHalluc (from TruthfulQA, general factual QA [33]), MedHalluc (from MedQA, clinical narratives and PubMedQA [34], eSNLI_Halluc (from eSNLI, inference with injected inconsistencies [35], and HotPotQA [36]. For each source, we convert items into question–answer (QA) style pairs [37] and then inject controlled inconsistencies by appending randomly sampled, semantically incompatible attributes (facts, circumstances, symptoms). These perturbations create positive "hallucination" cases; unmodified items

serve as negatives. Our focus is hallucination detection for model answers using four logical assessment methods as validators.

Table 9 shows F1 of hallucination detection for our datasets and our development of ALP. One can observe that Full D-ALP outperforms Abductive RAG system on HotPotQA dataset.

Table 9. Hallucination detection performance.

Dataset	Baseline ALP	ProbALP	D-ALP-Depth	D-ALP-Rel	Full D-ALP	Abductive RAG
TruthfulHalluc	0.63	0.66	0.70	0.72	**0.79**	
MedHalluc	0.63	0.68	0.71	0.75	**0.83**	
eSNLI_Halluc	0.60	0.68	0.71	0.72	**0.77**	
HotPotQA	0.65	0.64	0.68	0.72	**0.80**	0.75
Process-Control Hallucination	0.57	0.55	0.60	0.62	**0.72**	
Average	0.63	0.66	0.70	0.73	0.80	

Full D-ALP achieved 12–18 percentage points improvement over the baseline across datasets, confirming that discourse guidance improves abductive inference precision.

We observe in Table 10 that discourse structure pruned peripheral hypotheses and shortened inference time by over 20% compared to the baseline. Best performance is shown in bold.

Table 10. Efficiency of Full D-ALP in comparison with reduces ALPs.

System	Avg. Reasoning Time (s)	Search Space Reduction (%)
Baseline ALP	1.00	–
ProbALP	1.35	–
D-ALP-Depth	0.88	−12
D-ALP-Rel	0.82	−18
Full D-ALP	**0.79**	**−21**

We proceed to evaluation of logical consistency in Table 11. Best performance is shown in bold.

Table 11. Logical consistency.

System	Defeated Hypotheses (%)
Baseline ALP	19
ProbALP	15
D-ALP-Depth	13
D-ALP-Rel	10
Full D-ALP	**7**

Discourse-aware reasoning reduced logically inconsistent explanations by roughly two-thirds relative to the baseline.

8.4. Ablation Study

Baseline is the plain abductive logic program without any discourse information. Δ *Accuracy* means how many percentage points higher the correct-explanation rate became after adding each feature. Δ *Consistency* means how many percentage points fewer logically self-contradictory or "defeated" explanations appeared. In Table 12:

- D-ALP-Depth (+7%, +6%)—when only the nucleus/satellite hierarchy is used, the system already improves moderately. Prioritizing nucleus clauses focuses reasoning on the central facts and avoids side-track hypotheses.
- D-ALP-Rel (+10%, +9%)—adding rhetorical-relation knowledge (Cause, Evidence, Contrast, etc.) helps more: the model learns what type of relation best supports an abductive link, giving stronger logical and discourse coherence.
- Full D-ALP (+15%, +12%)—combining both depth and relation cues yields the largest gain. The two features reinforce one another: hierarchy tells the engine *which* statements matter most, relation type tells it *how* they connect.

Table 12. Ablation study of discourse features.

Variant	Δ *Accuracy* vs. Baseline (%)	Δ *Consistency* vs. Baseline (%)
D-ALP-Depth	+7	+6
D-ALP-Rel	+10	+9
Full D-ALP	**+15**	**+12**

8.5. Human Evaluation

A user study with 36 participants was conducted to assess explanation interpretability (Table 13). Participants included 12 engineers, 12 AI researchers, and 12 general users. Each participant reviewed 20 explanations per dataset (2160 total ratings). Explanations were rated on clarity, coherence, and trust (1 = poor, 5 = excellent). Best performance is shown in bold.

Table 13. Assessment of explanation.

System	Clarity	Coherence	Trust
Baseline ALP	3.1	2.9	2.8
ProbALP	3.3	3.0	3.0
D-ALP-Depth	4.0	3.8	3.9
D-ALP-Rel	4.2	4.1	4.1
Full D-ALP	**4.6**	**4.5**	**4.4**

Full D-ALP explanations were perceived as clearer and more coherent. Participants described them as “human-like,” “logical,” and “educational.”

Trust calibration was measured as the change in self-reported confidence before and after viewing explanations (Table 14). Best performance is shown in bold.

Table 14. Trust calibration.

System	Trust Before	Trust After	Δ (Calibrated Trust)
Baseline ALP	0.58	0.65	+0.07
ProbALP	0.57	0.67	+0.10
Full D-ALP	**0.55**	**0.78**	**+0.23**

Trust rose by 23% when discourse-structured reasoning was displayed, indicating higher user confidence grounded in transparency.

Alignment between system-generated and human discourse trees was also measured using F1 for nuclearity and rhetorical relations (Table 15).

Table 15. Alignment between machine and human discourse.

Dataset	Nucleus F1	Relation F1
TruthfulHalluc	0.73	0.69
MedHalluc	0.77	0.72
eSNLI_Halluc	0.70	0.65
Average	**0.73**	**0.69**

Higher alignment correlated strongly (r = 0.68) with human trust ratings. The results show that incorporating discourse structure enhances both reasoning performance and user understanding (Table 16). Discourse relations prioritize central claims and down-weight peripheral elaborations, leading to process control commands that are both logically precise and narratively coherent. Logical accuracy improved by up to 18%, while human clarity and trust rose by nearly 50%. The presence of rhetorical markers such as "because" and "however" increased users' sense that the model was reasoning transparently.

Table 16. Assessment for reasoning performance and user understanding.

Evaluation Aspect	Best System (Full D-ALP)	Improvement Over Baseline
Logical Accuracy	0.80 (average)	+16%
Runtime Efficiency	0.79 s/inference	−21%
Consistency	7% defeated	−63%
Human Clarity	4.6/5	+48%
Human Trust	4.4/5	+57%
Trust Calibration	+0.23	+0.16

Error analysis indicated that residual weaknesses stemmed from misclassification of rhetorical relations, particularly confusing Condition and Evidence, and from underestimation of multi-nuclear claims. Nonetheless, D-ALP consistently outperformed all baselines in accuracy, efficiency, and human trust.

Discourse-based abductive logic programming delivers process control commands that are both computationally efficient and intuitively meaningful. The framework advances explainable AI by integrating rhetorical structure into logic reasoning, yielding outcomes that are verifiable, interpretable, and trusted by humans. Future evaluations will extend to adaptive weighting learned from user feedback, cross-domain transfer to legal and educational corpora, and longitudinal studies of sustained trust.

8.6. Counter-Abduction and Hallucination Mitigation

An additional evaluation focused on how counter-abduction enhances D-ALP's capacity to detect and correct hallucinations in process control. Counter-abduction is defined here as the process of generating and testing *alternative hypotheses* that could back up the same control commands but contradict the current explanation. When integrated into D-ALP, counter-abduction acts as a built-in hallucination filter, comparing competing explanatory paths and identifying cases where the main abductive hypothesis lacks sufficient discourse or logical support.

To test this mechanism, the system was extended to compute for each hypothesis H a counter-hypothesis H' that minimizes overlap in supporting discourse nodes. A hypothesis was flagged as *hallucinatory* if its counter-hypothesis achieved equal or higher discourse-weighted plausibility (Table 17).

Adding counter-abductive reasoning increased hallucination detection accuracy by approximately 6 percentage points across datasets. The improvement was most pronounced

in narratives with ambiguous causal or evidential cues, where rival explanations helped expose unsupported assertions.

Table 17. Evaluation of counter-abduction.

Dataset	Baseline ALP Hallucination Accuracy	Full D-ALP (No Counter-Abduction)	D-ALP + Counter-Abduction
TruthfulHalluc	0.64	0.79	**0.86**
MedHalluc	0.66	0.83	**0.88**
eSNLI_Halluc	0.63	0.77	**0.84**
Average	**0.64**	**0.80**	**0.86**

Participants in a follow-up evaluation (N = 20) were shown pairs of system explanations—one original and one with counter-abductive contrast. Average trust in the system's factual grounding increased from 4.1 to 4.5 on the five-point scale. Annotators reported that explicitly seeing "why not-X" arguments enhanced their confidence that the reasoning engine could self-check for over-interpretation.

Qualitative review indicated that counter-abduction excels in identifying discourse-peripheral claims (often satellites or elaborations) that were incorrectly promoted to nucleus status by LLMs (Table 18). By generating rival hypotheses anchored in contrasting discourse relations—such as *Contrast*, *Condition*, or *Exception*—the system effectively invalidates spurious explanations before presentation to the user.

Table 18. Contribution of counter-abduction.

Evaluation Metric	Full D-ALP	D-ALP + Counter-Abduction
Hallucination Detection F1	0.81	**0.87**
False Positive Rate	0.14	**0.09**
Human Trust in Factuality	4.1	**4.5**

These findings demonstrate that counter-abduction provides a critical verification layer complementing the discourse-weighted abductive process. Together, they yield an interpretable mechanism for hallucination suppression that aligns with human reasoning: first hypothesize (abduction), then challenge (counter-abduction), and finally accept only the explanation that survives logical and rhetorical scrutiny.

9. Practical Deployment

The proposed abduction-based hallucination detection framework is not intended to operate inside the high-frequency inner control loop, but rather as a semantic verification layer integrated into supervisory and decision-support levels of a closed-loop process control architecture. In a standard hierarchy, the real-time control layer (e.g., PID or MPC) remains fully responsible for actuation and constraint enforcement, while the abductive verifier interfaces with LLM components that generate diagnoses, explanations, or recommendations. LLM outputs are treated as hypotheses, which are then evaluated against process knowledge, integrity constraints, and observed data using abductive and counter-abductive reasoning before being surfaced to operators or passed to supervisory logic (Figure 5).

In real-time settings, the framework operates event-driven rather than continuously. Verification is triggered when (i) an LLM proposes a fault explanation, root cause, or mitigation strategy; (ii) a recommendation would modify set-points, safety margins, or

operating modes; or (iii) anomalous conditions are detected that require human interpretation. Because abductive verification is applied selectively and at supervisory time scales (seconds to minutes), its computational overhead does not interfere with real-time control performance. The output of the verifier is a structured validation signal—accepted, rejected, or revised hypothesis—along with explicit justification or competing explanations.

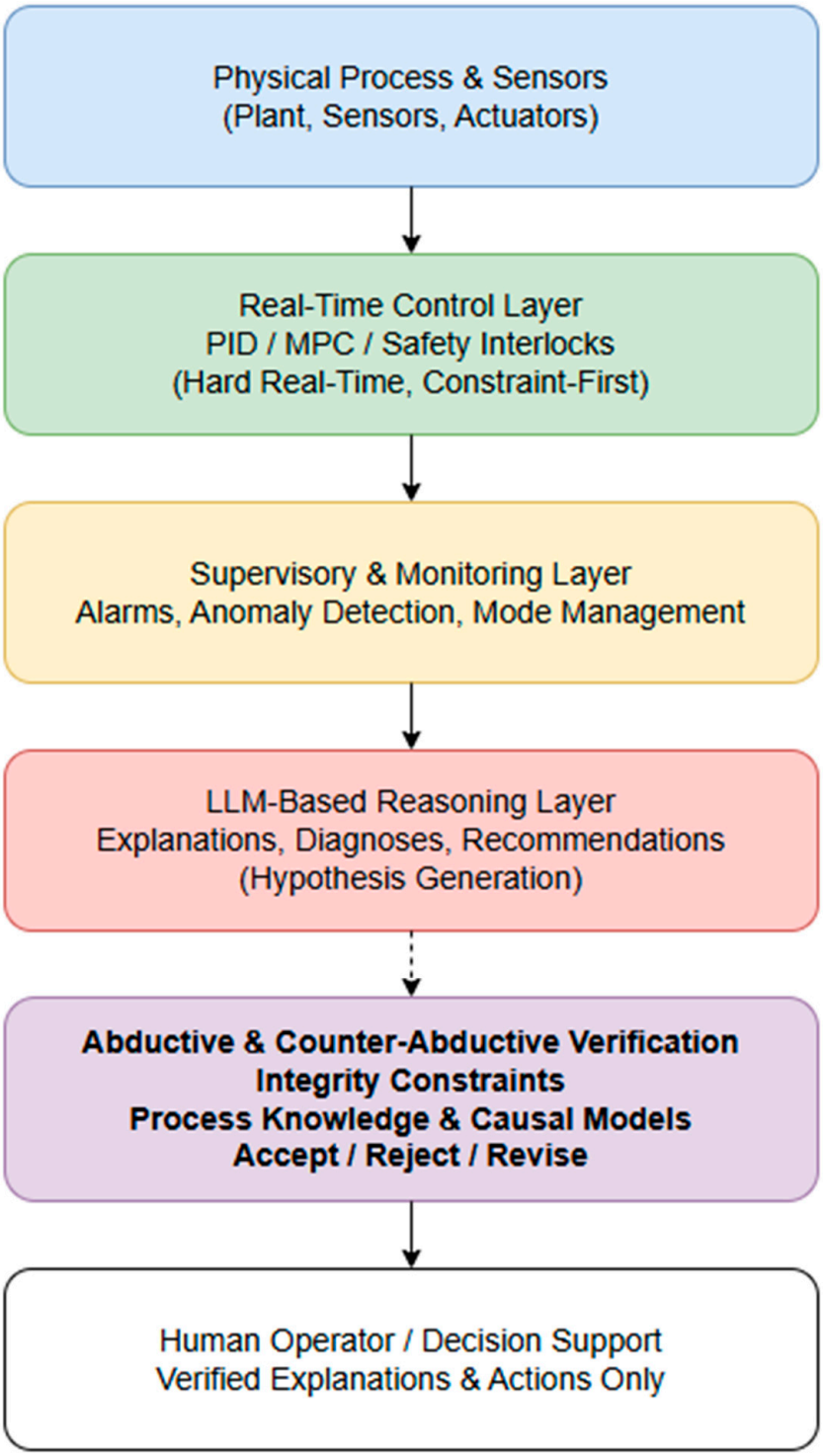


Figure 5. Integration of abduction-based hallucination detection into a closed-loop process control architecture. Real-time constrained controllers operate independently of LLM components, while abductive verification functions as a supervisory semantic safety layer that validates LLM-generated explanations and recommendations before they influence operators or supervisory decisions.

Usage scenarios are as follows:

1. Online monitoring and diagnosis. In monitoring applications, LLMs may summarize alarms, interpret multivariate trends, or propose fault diagnoses. The abductive verifier evaluates whether these interpretations are consistent with process constraints, causal dependencies, and available measurements. Hallucinated diagnoses—such as unsupported fault attributions or impossible causal chains—are flagged before being presented to operators, reducing cognitive overload and false confidence.
2. Decision support and supervisory control. In supervisory settings, LLMs may suggest operational adjustments, maintenance actions, or recovery strategies. Abductive verification checks whether such recommendations are supported by evidence and whether alternative explanations better explain the observed process behavior. Only verified recommendations are forwarded to supervisory controllers or operators, ensuring that downstream constrained controllers are not guided by semantically invalid inputs.
3. Human–AI collaboration and incident analysis. During abnormal events or post-incident reviews, the framework supports transparent human–AI interaction by exposing why a generated explanation is accepted or rejected. Counter-abductive reasoning makes implicit assumptions explicit, enabling operators and engineers to assess credibility rather than relying on fluent narratives. This strengthens trust without granting the LLM authority over control actions.

Software Implementation

The proposed framework is implemented using a modular neuro-symbolic pipeline that combines large language models with symbolic reasoning components. Logical reasoning and abductive inference are implemented in SWI-Prolog (v9.x), chosen for its mature support of meta-interpreters, constraint handling (CHR), and probabilistic extensions. Probabilistic abductive reasoning is realized using ProbLog, enabling uncertainty-aware hypothesis ranking and defeat scoring.

Discourse analysis is performed in two modes depending on scale. For small-scale, high-fidelity experiments, GPT-based discourse parsing is used to extract EDUs and rhetorical relations. For large-scale evaluation, a lightweight implementation based on the RST discourse parser [26] is employed to improve efficiency while preserving nucleus–satellite structure. Discourse trees are mapped to logical literals with associated weights and passed to the abductive solver.

The LLM component (GPT-5.1 in evaluation) is accessed via API and used exclusively for generation, not verification. All hallucination detection, counter-abduction, and ranking are performed externally in the symbolic layer, ensuring a black-box integration independent of model internals. Experiments were run on standard Linux servers; no specialized hardware (e.g., GPUs) is required for the reasoning layer. Configuration files specify abducibles, integrity constraints, discourse weights, and scoring parameters to ensure reproducibility across datasets and domains.

10. Related Work and Discussions

Although hallucination research has primarily focused on natural language tasks, the underlying phenomena—producing coherent but unsupported explanations—have clear analogs in industrial diagnostics and process control. Recent work has begun to explore these risks. FaultExplainer [38] demonstrates that LLMs used for fault diagnosis may generate plausible but incorrect mechanistic narratives, exposing the need for structured verification in safety-critical domains. Traditional fault detection and diagnosis (FDD) research provides a complementary foundation: model-based and data-driven

systems [39,40] emphasize consistency between observations, physical models, and causal pathways, thereby offering principles for identifying hallucination-like reasoning errors.

Explainable AI (XAI) for industrial systems further highlights the challenge of ensuring that diagnostic outputs remain grounded in sensor reality. Jang et al. [41] show that attribution-based explanations may fail to reflect true causal drivers, a form of mismatch analogous to hallucination in generative models. Work on rule induction for automated FDD [42] similarly notes that AI-generated rules may diverge from physical constraints unless tightly validated.

At the same time, research in NLP provides conceptual and methodological tools for detecting unsupported reasoning—hallucination taxonomies [17], verification-based prompting, and self-evaluation frameworks [43] highlight mechanisms through which LLMs can self-correct or expose faulty intermediate steps. While these studies do not address industrial processes directly, they motivate the need for domain-specific verification frameworks, such as abductive and counter-abductive reasoning, that integrate physical constraints and discourse evidence.

Taken together, these lines of work indicate a growing recognition that detection of ungrounded explanations—whether in language or process control—requires combining model-based reasoning, evidential validation, and structured hypothesis testing. Our work builds on this insight by proposing a unified abductive framework tailored to process-control hallucinations, where incorrect causal explanations can have significant operational and safety implications.

In abductive logic programming [22], the structure of the inference cycle:

1. Abductive generation: The system proposes candidate explanations—hypotheses that, when combined with background knowledge, entail the observed data.
2. Deductive testing: The implications of each hypothesis are deduced and checked against constraints.
3. Inductive evaluation: Empirical or probabilistic measures assess which hypotheses remain consistent with evidence.

Here, the availability constraint manifests through the space of abductive hypotheses the system can construct—defined by its symbolic vocabulary, its logical rules, and its computational resources. Thus, the inferential behavior of an ALP system corresponds not to an ideal IBE, which presupposes access to all possible explanations, but to an IBAE process bounded by representational and algorithmic feasibility.

Inductive inferences in the narrow sense are well-investigated, but their inferential power is limited. With their help it is possible to reason from regularities observed in the past to unobserved or future instances of these regularities, but it is not possible to infer conclusions containing new concepts expressing un observed properties that are not contained in the premises. These inferences are conceptually creative. They are needed in science whenever one reasons from the observed phenomena to theoretical concepts. Theoretical concepts in science describe unobservable properties (e.g., electric forces) or structures (e.g., electrons) that explain the observed phenomena in a unified way. It was an important insight of the post-positivistic philosophy of science that these theoretical concepts cannot be reduced to observable concepts via chains of definitions [29]. Thus the justification of explanations introducing theoretical concepts cannot be based on conceptual analysis, but must take the form of an ampliative inference.

Modern neuro-symbolic AI systems [44] face similar epistemic limitations. Even when neural networks are used to expand the hypothesis space by generating candidate patterns or latent variables, the symbolic reasoning layer can only evaluate those that fit within its logic schema. The model therefore performs an approximation to abduction, balancing between *expressive generativity* (neural) and *logical evaluability* (symbolic). This interplay

mirrors the philosophical IBAE trade-off: the system can only infer the best explanation among those currently expressible and computationally tractable.

Furthermore, the evaluation criterion—what counts as the "best" explanation—must also be contextually defined. In philosophy, explanatory virtues include simplicity, coherence, and unification [45]; in AI, analogous metrics include likelihood, posterior probability, minimality, or information gain. As in human inquiry, a system's "rational acceptability" depends on whether its best available explanation satisfies the methodological standards relevant to its domain—e.g., causal adequacy in science, logical consistency in expert systems, or interpretability in explainable AI.

HaluCheck is introduced as a visualization framework for assessing and prominently displaying the likelihood of hallucination in model outputs [10]. It allows users to select among multiple LLMs, providing flexibility for different tasks and preferences. The system integrates a diverse set of hallucination-evaluation metrics, enabling users to compute and compare likelihood scores using alternative methods. By allowing users to switch between these evaluators, HaluCheck supports experimentation with different assessment strategies and helps identify the most effective approach for a given use case.

Within the abductive framework developed in this paper, visualization plays an especially important role. Abductive and counter-abductive reasoning naturally generate sets of competing hypotheses, partial explanations, defeaters, and discourse-based evidence links. These structures are difficult to interpret when presented linearly, but they become immediately understandable when rendered visually. A visualization system such as HaluCheck can display the abductive reasoning graph—showing which evidence nodes support the main hypothesis, which discourse relations strengthen or weaken it, and where counter-hypotheses expose possible hallucinations. This makes the internal logic of hallucination detection transparent: users can see when an explanation relies on low-weight satellite information, when a rival hypothesis achieves a better discourse-weighted cost, or when inconsistency triggers a defeat. Visualizing these abductive and counter-abductive interactions thus strengthens interpretability, enables human verification of the hallucination score, and supports calibrated trust by revealing why a response is flagged as hallucinated rather than providing a numerical score alone.

Post-LLM Generation Verification

The concern regarding post-generation verification versus constraint-first synthesis is valid from a classical process control perspective, where embedding physical and operational constraints directly into control generation is essential for real-time safety guarantees. However, the proposed abductive verification framework is intentionally positioned at a different layer of the control stack. Constraint-first approaches, such as MPC and safety filters, ensure feasibility with respect to known constraints but implicitly assume that upstream reasoning—diagnoses, explanations, and recommendations—is semantically correct. LLM-generated outputs may satisfy all explicit constraints while remaining causally unsupported or incorrect. Abductive and counter-abductive reasoning specifically target these epistemic failures by evaluating whether a generated hypothesis is actually warranted by process evidence.

A key advantage of post-generation abductive verification is its model-agnostic and non-intrusive nature. Embedding detailed process constraints directly into LLM generation requires tight coupling to domain-specific formalisms and repeated re-engineering as models or constraints evolve, which is difficult in regulated industrial environments. In contrast, abductive verification operates as an external semantic checker that can be applied across different LLMs and deployments without modifying existing, certified control logic.

This makes the approach particularly suitable for incremental adoption in industrial settings where established constrained control architectures must remain intact.

Finally, concerns about computational overhead and latency are mitigated by the intended scope of application. The framework is not designed for high-frequency inner-loop control, but for supervisory, diagnostic, and decision-support tasks, where reaction times are significantly longer. In these contexts, the cost of post-generation verification is small relative to the risk of acting on hallucinated explanations or recommendations. Moreover, verification can be selectively triggered only when outputs affect control-relevant decisions. Viewed in this way, post-generation abductive verification complements constraint-first control by providing a semantic safety layer that enhances trust, transparency, and robustness in LLM-enabled process control systems.

11. Conclusions

The discourse-aware abductive framework introduced in this work provides a principled foundation for constructing and verifying complex explanations generated by large language models. By integrating abductive inference with rhetorical structure analysis, the approach enables systems to distinguish central, hypothesis-bearing content from peripheral or contextual material, thereby strengthening both explanatory precision and hallucination detection. The value of this integration is evident across multiple application domains.

Counter-abduction is thus a foundational component of hallucination-resistant neuro-symbolic reasoning. By positioning rival explanations as defeaters of LLM-generated CoTs, counter-abductive reasoning transforms narrative reasoning into a competitive, evidence-driven process grounded in logic and discourse structure. This provides a unified theoretical and computational basis for hallucination detection and correction across medical analysis, legal reasoning, scientific interpretation, and general-purpose CoT verification.

In medical narratives, weighting discourse nuclei over satellite descriptions allows the system to focus abductive diagnosis generation on patient-relevant complaints rather than tangential remarks, improving causal hypothesis extraction. In legal reasoning, the framework supports more transparent argument evaluation by giving precedence to claims occurring in the conclusion or main argument segments while attenuating the influence of background information. In scientific writing, it enhances the identification of robust causal explanations by prioritizing claims derived from results and discussion sections over speculative or forward-looking commentary. Finally, in LLM verification, discourse-aware abductive logic programming offers a structured mechanism for identifying hallucinations: statements originating in low-weight, peripheral text segments can be selectively discounted, while central claims undergo rigorous consistency checking.

Taken together, these applications demonstrate that combining abductive reasoning with discourse structure provides a versatile and effective method for improving reasoning fidelity, ensuring interpretability, and increasing trust in neuro-symbolic systems across diverse high-stakes domains.

Advantages are:

- Increases interpretability: abductive hypotheses are justified by discourse roles.
- Improves precision: ignores peripheral text when generating explanations.
- Enables alignment with human reasoning: since humans emphasize nuclei when forming explanations.
- Supports hallucination detection: contradictions in nucleus-derived claims outweigh peripheral inconsistencies.

"Counter-abduction strength of confirmation metrics dialogue" refers to a structured, interactive process where

1. Abductive reasoning proposes initial explanations for observed evidence.
2. Counter-abduction introduces competing explanations.
3. Confirmation metrics quantitatively assess how well the evidence supports each hypothesis.
4. Dialog facilitates the comparison and discussion of these assessments to arrive at the most plausible explanation.

This framework is powerful in any scenario requiring rigorous evaluation of competing hypotheses, ensuring that conclusions are well-supported by evidence. It combines logical reasoning, probabilistic assessment, and collaborative discussion to navigate complex, uncertain situations effectively.

Our evaluation demonstrates that incorporating discourse structure and counter-abductive reasoning significantly improves both the logical validity and operational credibility of AI-generated diagnostics in chemical process control. Discourse-weighted Abductive Logic Programming (D-ALP) not only produces fault explanations that are consistent with process physics and observed signals, but also actively tests these explanations against competing fault hypotheses, thereby exposing and reducing hallucinated causal narratives. The results indicate that explanation robustness—rather than surface plausibility alone—is critical for reliable decision support in safety-critical chemical systems. More broadly, abductive hallucination detection can be deployed as a verification layer atop existing white-box, gray-box, and black-box diagnostic approaches [46,47], allowing it to complement data-driven, model-based, and hybrid monitoring techniques [48]. Together, these findings position discourse-aware abductive reasoning as a practical foundation for verifiable and trustworthy neuro-symbolic AI in industrial process monitoring and control.

Safety-aware planning and constrained control methods, although not dependent on LLMs, address a closely related fundamental objective: guaranteeing the feasibility, safety, and regulatory compliance of decisions in process control environments. Constraint-first synthesis techniques—such as model predictive control, reference governors, and formal safety filters—achieve this goal by explicitly restricting the admissible control space and enforcing physical and operational limits by construction [14,16]. These approaches are highly effective at preventing unsafe control actions but implicitly assume that upstream inputs—process diagnoses, explanations, objectives, or operator guidance—are themselves valid and well grounded in process knowledge. As a result, they do not directly address epistemic failures in reasoning or explanation that may arise when LLM-based components are introduced into the control workflow.

In contrast, the abductive verification framework proposed in this work operates at the semantic reasoning layer, targeting hallucinations in explanations, diagnoses, and recommendations generated by LLM-assisted process control systems. Rather than synthesizing control actions under constraints, our method evaluates whether a proposed claim is logically supported by process knowledge, causal structure, and observed evidence, and exposes unsupported outputs through counter-abductive reasoning [17,21]. This distinction clarifies the novelty and scope of the contribution: abductive verification complements, rather than replaces, constraint-first control by guarding against fluent but incorrect reasoning that may pass syntactic or feasibility checks yet remain semantically invalid. By integrating abductive hallucination detection as a verification layer above established constrained control architectures, the proposed approach strengthens trust, interpretability, and safety in LLM-enabled process control without disrupting existing control enforcement mechanisms.

Limitations

While the proposed abductive and counter-abductive framework demonstrates clear advantages in detecting explanation-level hallucinations in process control, several limitations should be acknowledged. First, the current evaluation relies primarily on synthetic and semi-synthetic process scenarios, which, although physically grounded and carefully constructed, cannot fully capture the variability, noise, and undocumented contingencies present in real industrial plants. Second, the quality of abductive inference depends on the completeness and correctness of the underlying process ontology; missing causal links or oversimplified fault models may lead to incomplete hypothesis spaces or false defeasibility. Third, discourse parsing—particularly when using high-capacity LLMs—introduces computational overhead that may limit real-time deployment in high-frequency monitoring settings, although lighter parsers mitigate this at some cost to granularity. Finally, while the framework is designed to operate as a verification layer atop white-box, gray-box, and black-box diagnostic systems, its performance is sensitive to the quality of upstream signals and explanations. Addressing these limitations will require broader validation on real plant data, expansion of domain ontologies, and further optimization for online industrial use.

Author Contributions: Conceptualization, B.G.; Methodology, B.G.; Software, B.G. and A.R.; Validation, B.G.; Formal analysis, B.G. and A.R.; Investigation, B.G.; Data curation, B.G.; Writing—original draft, B.G.; Writing—review and editing, A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Code is available at https://github.com/bgalitsky/halluc_in_health/tree/master/abduction (accessed on 12 January 2026). The original data presented in the study are openly available in: https://github.com/bgalitsky/halluc_in_health/tree/master/process_control (accessed on 12 January 2026).

Acknowledgments: During the preparation of this manuscript/study, the authors used GPT 5.1 and Qwen 3 for the purposes of English grammar improvement. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: Author Boris Galitsky was employed by the Knowledge Trail Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Qin, S.J.; Badgwell, T.A. A survey of industrial model predictive control technology. *Control Eng. Pract.* **2003**, *11*, 733–764. [CrossRef]
2. Galitsky, B. A neuro-symbolic system for distributed chemical engineering simulations. In *Applications of Neuro-Symbolic AI*; Springer: Cham, Switzerland, 2026; *in press*.
3. Venkatasubramanian, V. A review of process fault detection and diagnosis: Part I–III. *Comput. Chem. Eng.* **2003**, *27*, 293–346. [CrossRef]
4. Lee, J.; Bagheri, B.; Kao, H.A. A cyber-physical systems architecture for Industry 4.0-based manufacturing systems. *Manuf. Lett.* **2015**, *3*, 18–23. [CrossRef]
5. Peres, R.S.; Jia, X.; Lee, J.; Sun, K.; Colombo, A.W.; Barata, J. Industrial artificial intelligence in industry 4.0—Systematic review, challenges and outlook. *IEEE Access* **2020**, *8*, 220121–220139. [CrossRef]
6. Decardi-Nelson, B.; Alshehri, A.S.; Ajagekar, A.; You, F. Generative AI and process systems engineering: The next frontier. *Comput. Chem. Eng.* **2024**, *187*, 108723. [CrossRef]
7. Azaria, A.; Mitchell, T. The internal state of an LLM knows when it's lying. *arXiv* **2023**, arXiv:2304.13734.
8. Su, W.; Wang, C.; Ai, Q.; Hu, Y.; Wu, Z.; Zhou, Y.; Liu, Y. Unsupervised real-time hallucination detection based on the internal states of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*; Ku, L.W., Martins, A., Srikumar, V., Eds.; Association for Computational Linguistics: Bangkok, Thailand, 2024; pp. 14379–14391.

9. Varshney, N.; Yao, W.; Zhang, H.; Chen, J.; Yu, D. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv* **2023**, arXiv:2307.03987. [CrossRef]
10. Heo, S.; Son, S.; Park, H. Halucheck: Explainable and verifiable automation for detecting hallucinations in llm responses. *Expert Syst. Appl.* **2025**, *272*, 126712. [CrossRef]
11. Kossen, J.; Han, J.; Razzak, M.; Schut, L.; Malik, S.; Gal, Y. Semantic entropy probes: Robust and cheap hallucination detection in LLMs. *arXiv* **2024**, arXiv:2406.15927. [CrossRef]
12. Galitsky, B. Improving open domain content generation by text mining and alignment. In *AI for Health Applications and Management*; Galitsky, B., Goldberg, S., Eds.; Elsevier: Amsterdam, The Netherlands, 2021.
13. Shi, X.; Xue, S.; Wang, K.; Zhou, F.; Zhang, J.; Zhou, J.; Tan, C.; Mei, H. Language Models Can Improve Event Prediction by Few-Shot Abductive Reasoning. *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 29532–29557.
14. Rawlings, J.B.; Mayne, D.Q. *Model Predictive Control: Theory and Design*; Nob Hill Publishing: San Francisco, CA, USA, 2009.
15. Åström, K.J.; Hägglund, T. *PID Controllers: Theory, Design, and Tuning*; ISA: Singapore, 1995.
16. Gilbert, E.G.; Kolmanovsky, I.; Tan, K.T. Discrete-time reference governors and the nonlinear control of systems with state and control constraints. *Int. J. Robust Nonlinear Control* **1995**, *5*, 487–504. [CrossRef]
17. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Madotto, A.; Fung, P. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.* **2023**, *55*, 12. [CrossRef]
18. Rawlings, J.B.; Meadows, E.S.; Muske, K.R. Nonlinear model predictive control: A tutorial and survey. *IFAC Proc. Vol.* **1994**, *27*, 185–197. [CrossRef]
19. Peirce, C.S. Illustrations of the Logic of Science. Popular Science Monthly. 1878. Volume 13. Available online: https://en.wikisource.org/wiki/Popular_Science_Monthly/Volume_13/August_1878/Illustrations_of_the_Logic_of_Science_VI (accessed on 12 January 2026).
20. Peirce, C.S. Lectures on Pragmatism. In *Collected Papers of Charles Sanders Peirce*; Hartshorne, C., Weiss, P., Eds.; Harvard University Press: Cambridge, MA, USA, 1903; Volume 5.
21. Harman, G. The inference to the best explanation. *Philos. Rev.* **1965**, *74*, 88–95. [CrossRef]
22. Kakas, A.C.; Mancarella, P. Generalized Abduction. *J. Log. Comput.* **1990**, *1*, 389–407.
23. Swi-Prolog. 2025. Available online: https://www.swi-prolog.org/ (accessed on 12 January 2026).
24. Christiansen, H. Executable specifications for hypothesis-based reasoning with Prolog and Constraint Handling Rules. *J. Appl. Log.* **2009**, *7*, 341–362. [CrossRef]
25. Galitsky, B.; Rybalov, A. Extending large language model capabilities beyond reasoning. In *Healthcare Applications of Neuro-Symbolic Artificial Intelligence*; Elsevier: Amsterdam, The Netherlands, 2025; Chapter 4; pp. 81–107.
26. Jansen, P.; Surdeanu, M.; Clark, P. Discourse complements lexical semantics for non-factoid answer reranking. In Proceedings of the 52th Annual Meeting ACL, Baltimore, MD, USA, 22–27 June 2014. [CrossRef]
27. Kakas, A.C.; Kowalski, R.A.; Toni, F. Abductive logic programming. *J. Log. Comput.* **1992**, *2*, 719–770. [CrossRef]
28. Haig, B.D. *Investigating the Psychological World: Scientific Method in the Behavioral Sciences*; MIT Press: Cambridge, MA, USA, 2014.
29. Carnap, R. *The Logical Foundations of Probability*, 2nd ed.; University of Chicago Press: Chicago, IL, USA, 1962.
30. Crupi, V.; Tentori, K.; González, M. On Bayesian measures of evidential support. *Philos. Sci.* **2007**, *74*, 229–252. [CrossRef]
31. Galitsky, B. Identifying large language model hallucinations in health communication. In *Healthcare Applications of Neuro-Symbolic Artificial Intelligence*; Elsevier: Amsterdam, The Netherlands, 2025; Chapter 8; pp. 283–329.
32. Lin, S. Abductive Inference in Retrieval-Augmented Language Models: Generating and Validating Missing Premises. *arXiv* **2025**, arXiv:2511.04020v1. [CrossRef]
33. Lin, S.; Hilton, J.; Evans, O. TruthfulQA: Measuring How Models Mimic Human Falsehoods. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; Volume 1, pp. 3214–3252. [CrossRef]
34. Jin, D.; Pan, E.; Oufattole, N.; Weng, W.-H.; Fang, H.; Szolovits, P. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *arXiv* **2020**, arXiv:2009.13081. [CrossRef]
35. Camburu, O.-M.; Rocktäschel, T.; Lukasiewicz, T.; Blunsom, P. e-SNLI: Natural language inference with natural language explanations. *arXiv* **2018**, arXiv:1812.01193. [CrossRef]
36. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, 31 October–4 November 2018; ACL Anthology. Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 2369–2380. [CrossRef]
37. Rasool, Z.; Kurniawan, S.; Balugo, S.; Barnett, S.; Vasa, R.; Chesser, C.; Hampstead, B.M.; Belleville, S.; Mouzakis, K.; Bahar-Fuchs, A. Evaluating LLMs on document-based QA: Exact answer selection and numerical extraction using CogTale dataset. *Nat. Lang. Process. J.* **2024**, *8*, 100083. [CrossRef]
38. Khan, A.; Nahar, R.; Chen, H.; Flores, G.E.C.; Li, C. FaultExplainer: Leveraging large language models for interpretable fault detection and diagnosis. *Comput. Chem. Eng.* **2025**, *199*, 109152. [CrossRef]

39. Venkatasubramanian, V. The promise of artificial intelligence in chemical engineering: Is it here, finally? *AIChE J.* **2019**, *65*, 466–478. [CrossRef]
40. Isermann, R. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*; Springer: Berlin/Heidelberg, Germany, 2006.
41. Jang, K.; Pilario, K.E.; Lee, N.; Moon, I.; Na, J. Explainable Artificial Intelligence for Fault Diagnosis of Industrial Processes. *IEEE Trans. Ind. Inform.* **2023**, *21*, 4–11. [CrossRef]
42. Fischer, J. AI-Powered Rule Generation for Automated Fault Detection and Diagnostics. Doctoral Dissertation, Technische Universität Wien, Vienna, Austria, 2025.
43. Miao, N.; Teh, W.Y.; Rainforth, T. Selfcheck: Using LLMs to zero-shot check their own step-by-step reasoning. *arXiv* **2023**, arXiv:abs/2308.00436.
44. Bader, S.; Hitzler, P. Dimensions of Neural-Symbolic Integration—A Structured Survey. *arXiv* **2005**, arXiv:cs/0511042.
45. Lipton, P. *Inference to the Best Explanation*, 2nd ed.; Routledge: London, UK, 2004.
46. Xie, L.; Liu, H.; Zeng, J.; Tang, X. A Survey of Calibration Process for Black-Box LLMs. *arXiv* **2024**, arXiv:2412.12767. [CrossRef]
47. Wu, J.; Dong, N.; Liu, F.; Yang, S.; Hu, J. Feature hallucination via Maximum A Posteriori for few-shot learning. *Knowl.-Based Syst.* **2021**, *225*, 107129. [CrossRef]
48. Galitsky, B.; Tsirlin, A. Step Wise Approximation of CBOW Reduces Hallucinations in Tail Cases. 2025. Available online: https://www.preprints.org/manuscript/202507.0670 (accessed on 12 January 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.