Building Trustworthy AI by Addressing its 16+2 Desiderata with Goal-Directed Commonsense Reasoning

Alexis R. Tudor[1] ALEXISRENEE1@GMAIL.COM
Yankai Zeng[1] YANKAI.ZENG@UTDALLAS.EDU
Huaduo Wang[1] HUADUO.WANG@UTDALLAS.EDU
Joaquín Arias[2] JOAQUIN.ARIAS@URJC.ES
Gopal Gupta[1] GUPTA@UTDALLAS.EDU
[1]University of Texas at Dallas, Texas, USA

[2]CETINIA, Universidad Rey Juan Carlos, Madrid, Spain

Abstract

Current advances in AI and its applicability have highlighted the need to ensure its trustworthiness for legal, ethical, and even commercial reasons. Sub-symbolic machine learning algorithms, such as the LLMs, simulate reasoning but hallucinate and their decisions cannot be explained or audited (crucial aspects for trustworthiness). On the other hand, rule-based reasoners, such as Cyc, are able to provide the chain of reasoning steps but are complex and use a large number of reasoners. We propose a middle ground using s(CASP), a goal-directed constraint-based answer set programming reasoner that employs a small number of mechanisms to emulate reliable and explainable human-style commonsense reasoning. In this paper, we explain how s(CASP) supports the 16 desiderata for trustworthy AI introduced by Doug Lenat and Gary Marcus (2023), and two additional ones: inconsistency detection and the assumption of alternative worlds. To illustrate the feasibility and synergies of s(CASP), we present a range of diverse applications.

1. Introduction

Although incredible advancements in artificial intelligence (AI) have been made, robustness in AI systems is still lacking. The recent AAAI study by Rossi & et al. (2025) found that a "majority of respondents (76%) assert that 'scaling up current AI approaches' to yield AGI is 'unlikely' or 'very unlikely' to succeed, suggesting doubts about whether current machine learning paradigms are sufficient for achieving general intelligence". Despite that, the predominant approach towards AI remains pouring more resources, such as data, training time, and GPUs, into generative AI in order to produce marginally better results. We argue that what is necessary is not larger and more robust neural networks, but rather the encoding of commonsense knowledge alongside a capable reasoner. In this paradigm, deep learning becomes a tool for a reasoner to interface with, not an entity in itself.

Researchers have agreed since the inception of AI that commonsense knowledge and reasoning are important for the field. There are two fundamental components in commonsense reasoning: the

This work is licensed under a Creative Commons Attribution International 4.0 License.

knowledge base, containing all the commonsense knowledge about the problem domain, and the commonsense reasoner, which processes that knowledge to draw new conclusions.

Sub-symbolic vs. symbolic AI: The first part, the knowledge base, almost universally poses a problem for modern deep learning methods. In deep learning, knowledge is extracted from huge amounts of data. It is impossible to perform any reasonable quality assurance at the required scale. This problem is being tackled by improving the ability to train on small datasets and through the use of services like Amazon's Mechanical Turk to label large quantities of data with inexpensive labor. Using data that is not labeled or evaluated for quality tends to lead to difficulty with reasoning in these systems. In some domains, a strong foundational knowledge base is almost inherently machine-usable; however, in others the translation from knowledge to reasoning is more difficult. Deep learning systems are incredibly robust to ingesting large quantities of data, but they do not reason and are prone to errors that would not be made by a system with sound commonsense knowledge. On the other hand, logic-based symbolic reasoners struggle with representing and efficiently processing large amounts of knowledge. A combination of deep learning and logic-based reasoning, a neuro-symbolic system, performs better on key metrics like explainability without sacrificing performance, see survey by Yu et al. (2023). Overall, systems that combine a robust knowledge base with a reasoner designed to utilize it, perform the best. The problem is far from solved though, as brittleness of the logic-based knowledge base and commonsense reasonng remains a challenge.

Cyc, a machine reasoner: One of the oldest attempts to use commonsense knowledge is the Cyc project. Cyc seeks to encode all commonsense knowledge into a knowledge base following its own proprietary ontology (Lenat, 1995). For four decades the Cyc knowledge base has grown, gaining hand-crafted and hand-categorized axioms of knowledge covering every imaginable subject and ambiguity in the hopes of using that knowledge to cultivate true human-like intelligence.

However, outside of Cycorp the reception of Cyc has been lukewarm. Cyc's philosophy has not pervaded AI research, possibly due to Cyc being a closed system, and deep learning-based AI research has moved to the forefront. Many of Cyc's critics use it as an example to prove that symbolic AI is doomed for failure. However, as we look to the next steps for AI beyond deep learning, other symbolic approaches solve some of its greatest weaknesses and show promise for the future. We posit that purely deep learning systems require symbolic intervention to improve their trustworthiness. Explainability is fundamentally important for AI systems that are trustworthy enough for critical use.

LLMs and Trustworthiness: Recent work by Lenat & Marcus (2023) discusses what is needed for truly trustworthy AI. The paper argues that Large Language Models (LLMs) are untrustworthy, unstable, and brittle, something that has continued to prove true as LLMs have become more and more ubiquitous. They posit that generative AI systems fail because they do not reason, but rather are very sophisticated pattern matching algorithms. They argue that LLMs would need to pair up with a system like Cyc to become trustworthy. However, Cyc is a complex and proprietary system.

Answer Set Programming: For symbolic reasoning systems to succeed, simplicity must be paramount. The answer set programming (ASP) paradigm has shown how commonsense knowledge can be represented and reasoned over in an elegant way (Brewka et al., 2011; Gelfond & Kahl, 2014; Gupta et al., 2022). Implementations of ASP such as the CLINGO system (Gebser et al., 2014), however, have a problem, namely, that they are based on grounding the program and then using a SAT solver

or a procedure inspired by SAT to find answer sets. Thus, they do not scale well and are primarily used to solve combinatorial problems rather than large knowledge-based reasoning problems. To alleviate this, we propose to model commonsense reasoning with goal-directed ASP engines, which do not ground programs and support constructive negation, giving rise to what we have termed **goal-directed commonsense reasoning**. In particular, we propose the use of s(CASP), of which we provide a brief review in Section 2.

Addressing 16+2 Desiderata: This paper provides a structured argument that s(CASP) plausibly satisfies the desiderata established by Lenat & Marcus (2023) as capabilities required by general AI to be trustworthy. We also discuss additional desiderata that a (goal-directed) commonsense reasoning system must satisfy: (i) being able to specify inconsistencies via global constraints, and (ii) being able to incorporate assumptions through multiple possible worlds. The s(CASP) system incorporates these additional desiderata. Given the simplicity of s(CASP), the fact that it is open-sourced, and that it can support all the desiderata, we present it as a publicly available system for building trustworthy AI in conjunction with machine learning. These 16+2 desiderata are reviewed in Section 3 with a brief description, related s(CASP) capabilities, and a discussion of related work. The main contribution of this paper is to organize the capabilities of s(CASP), and present some of its applications in terms of the desiderata for trustworthy AI.

2. Background: s(CASP)

s(CASP), by Arias et al. (2018), is a novel non-monotonic reasoner that evaluates Constraint Answer Set Programs without a grounding phase either before or during execution. s(CASP) supports predicates and thus retains logical variables (and constraints) both during the execution and in the answer sets. The operational semantics of s(CASP) rely on backward chaining, which is intuitive to follow and lends itself to generating explanations that can be translated into natural language (NL). The execution of an s(CASP) program is goal-directed (i.e., starts with a query), and returns partial stable models: the subsets of the stable models, defined by Gelfond & Lifschitz (1988), which include only the positive or negative literals necessary to support the initial query. Any computed bindings for unbound variables in the query are also returned. The s(CASP) system supports first order predicates and terms, constructive negation, and coinduction. More details on s(CASP) are not included due to lack of space, but they can be found elsewhere (Arias et al., 2018; Gupta et al., 2022). s(CASP) has been used for many applications related to the representation of commonsense reasoning (including commercial ones as in Forsante Oy (2024)). We outline the ones below that are crucial for this paper:

VECSR: Virtually Embodied Common Sense Reasoning (VECSR) by Tudor et al. (2025) is a system that accepts high-level tasks and uses s(CASP) to plan and execute those tasks in a context-aware fashion in VirtualHome, a fully-embodied simulated environment.

Reliable Chatbots: In the Amazon Alexa Socialbot Grand Challenge 4, Basu et al. (2021b) developed a conversational AI chatbot based on two natural language understanding systems by Basu et al. (2021a). Going further, Zeng et al. (2024) used LLMs to translate natural language into predicates (and vice versa) and employ commonsense reasoning based on s(CASP) to build a reliable task-bot as well as a socialbot to hold a social conversation with a human.

Event Calculus (EC): A reasoner on EC by Arias et al. (2022a) has been used by Hall et al. (2021) to model real-world avionics systems, verify timed properties, and identify gaps in system requirements, and by Vašíček et al. (2024) to model a medical device.
Justification in NL: Has been used by Arias et al. (2020) and Chen et al. (2016) to bring eXplainable Artificial Intelligence (XAI) principles to expert knowledge systems.
ILP systems: Inductive Logic Programming (ILP) systems generate logic programs from data. This includes the FOLD family of algorithms, the latest of which is FOLD-SE by Wang & Gupta (2024) that learns stratified answer set programs.
s(LAW): An administrative and judicial discretion reasoner by Arias et al. (2024a), which allows modeling of legal rules involving ambiguity to infer conclusions and provides natural language justifications for those conclusions.
Spatial reasoner: Implemented by Arias et al. (2022c), it models dynamic information and restrictions in Building Information Modeling (BIM) and paves the way to using logic-based methodologies such as model refinement.

3. Addressing 16+2 desiderata with s(CASP)

The 16 desiderata for trustworthy AI are described in detail by Lenat & Marcus (2023), and so this section will only briefly summarize them. Table 1 indicates how various AI systems accomplish the desiderata for reference. For each of the 16+2 desiderata, we provide examples of how the s(CASP) system fulfills them. In some applications used for illustration, we describe how we successfully combine machine learning and s(CASP)-based automated commonsense reasoning with a goal of achieving human-level intelligence at least for domain-specific tasks.

#01 Explanation: *Explainability is a commonly desired trait of AI, stating that any trustworthy AI must be able to explain any solution found. Its importance is reflected in the growing field of XAI.*

s(CASP): Goal-directed execution means that one poses queries that are answered by an SLD-resolution-like process, similar to Prolog, and therefore, a proof is explicitly constructed. This proof constitutes not only the explanation but also the justification of a given decision (positive or negative). In s(CASP), the justification for any query is the trace of the successful derivation, or in cases where the query fails, is the trace of the negated query. Moreover, the framework developed by Arias et al. (2020) allows the generation of justifications in natural language and navigable HTML files. Thus, explainability is native to s(CASP) and has been exploited in applications such as the physician advisor for chronic heart failure management by Chen et al. (2016) and by Forsante Oy (2025).

Discussion: One of the great weaknesses of deep learning systems is that they are unexplainable black boxes, where it is difficult to justify a solution. The closest LLMs get is that they can be asked "why" an answer was given, but the explanation provided may be for a different alternative solution or the justification itself may be a hallucination.

#02 Deduction: *A foundational aspect of trustworthy AI is its capacity for human-like deduction, encompassing logical reasoning (e.g., modus ponens), arithmetic operations, exhaustive search,*

Table 1: Desiderata support by Cyc, Deep-Learning, and s(CASP). Including s(CASP) features with citation.

		Cyc	DL	s(CASP)	Feature / Tool
#01	Explanation	✓	✗	✓	*Justification Trees*[1]
#02	Deduction	✓	!	✓	*Base s(CASP)*[2]
#03	Induction	!	!	✓	*Default Rules + FOLD*[3]
#04	Analogy	!	!	!	*Proposed LLM + s(CASP)*
#05	Abductive Reasoning	✓	✗	✓	*Abducibles*[2]
#06	Theory of Mind	✗	✓	✓	*Chatbot*[4]
#07	Quantifier-fluency	✗	✗	✓	*Goal-Directed Execution*[2]
#08	Modal-fluency	✓	✓	✓	*Deontic Logic*[5]
#09	Defeasibility	✓	✓	✓	*Default Rules + FOLD*[3]
#10	Pro and Con Arguments	✓	✗	✓	*ASP Multiple Worlds*[6]
#11	Contexts	✓	✗	✓	*VECSR*[7]
#12	Meta-knowledge and Meta-reasoning	✗	✗	✓	*Justification Trees*[1]
#13	Explicitly Ethical	!	✗	✓	*Constraints*[2]
#14	Sufficient Speed	✗	!	✓	*VECSR*[7]
#15	Sufficiently Lingual and Embodied	✗	✓	✓	*Chatbot*[4]
#16	Broadly and Deeply Knowledgeable	✗	!	✓	*FOLD, Chatbot, VECSR*[3,4,7]
#17	Inconsistency Detection	!	✗	✓	*Constraints*[2]
#18	Multiple Possible Worlds	!	✗	✓	*ASP Multiple Worlds*[6]

Note: ✓ means fully accomplished, ! means partially accomplished, and ✗ means non-achieved.
[1] Arias et al. (2020), [2] Arias et al. (2018), [3] Wang & Gupta (2024), [4] Zeng et al. (2024), [5] Gupta et al. (2025), [6] Gelfond & Lifschitz (1988), [7] Tudor et al. (2025)

and the identification of contradictions or redundancies. This includes understanding logical connectives, in particular recognizing different forms of negation, i.e., not being able to conclude p is different from being able to conclude p is false.

s(CASP): Deduction is the basis of symbolic systems based on logic. Given premises p and p $\Longrightarrow$ q, we deduce q. Suppose we are given the premises that Tweety is a bird, bird(tweety), and the formula $\forall$ X. bird(X) $\Longrightarrow$ flies(X). From these two premises, we can deduce that flies (tweety) holds, i.e., Tweety can fly. Constraint Programming adds declarative arithmetic processing to logic programming. The mortgage example by Holzbaur (1995) allows us to reason about the relation among the principal, P, the repayment rate, Mp, and the balance owing, B, i.e., the query ?– mortgage(P,12,,0.01,B,Mp) returns P = 6.14∗R + 0.38∗B. Finally, to deal with negation as failure (or default negation) when we lack evidence, the stable model semantics of Gelfond & Lifschitz (1988) is required, e.g., the rules p :– not q[1] and q :– not p, have two stable models {p} and {q}. The stable model semantics can be realized via answer set programming (ASP). ASP also incor-

1. P :– Q expresses the premise Q $\Longrightarrow$ P.

porates classical (or strong) negation in addition to default negation, e.g., −flies (X) :− penguin(X) means that X does not fly if it is a penguin (fly (X) is definitely false). Strong negation is used to give an explicit proof of a predicate's falsehood.
These features are integrated into s(CASP) and have been leveraged for various applications discussed earlier.

Discussion: Prior research has found that these deductive (along with inductive and abductive) reasoning skills are somewhat lacking in LLMs (Cheng et al., 2024; Abe et al., 2025). This is likely due in part to the fact that LLMs do not reason, but rather match patterns, and so are prone to confabulations. Cyc, on the other hand, provides mixed results. Cyc has over 1,000 reasoners for deductions that are each optimized for different kinds of knowledge, contexts, and solution finding. This provides a wide breadth of deduction ability, however there are disadvantages associated with having such a large number of reasoning engines. Namely, processing power required for running multiple reasoners, and overhead from meta-reasoners deciding which reasoners are needed. There is a trade-off between accuracy and speed associated with symbolic or neural methods. _____

#03 Induction: *Effective inductive reasoning is essential for navigating complex, uncertain environments. It involves generalizing from specific observations, such as inferring species traits, and making temporal projections where the probability of a fact holding over time often follows predictable decay curves (e.g., linear, Gaussian). Such reasoning supports adaptive decision-making, despite the inherent risk of error. This type of reasoning is the foundation for explanation-based learning, which seeks to learn more with less data based on inducing general rules from specific examples.*

s(CASP): Inductive Logic Programming (ILP), see survey by Cropper & Dumancic (2022), is a sub-field of machine learning that learns interpretable logic programs from labeled data and background knowledge. Unlike statistical models, ILP uses symbolic representations and logical inference to derive hypotheses that generalize observations in a human-understandable way. This makes ILP valuable in domains requiring transparency, prior knowledge integration, and formal reasoning, such as bio-informatics, legal systems, and explainable AI. The FOLD family of algorithms extends ILP by enabling the learning of default rules with exceptions, capturing common patterns of non-monotonic reasoning (Gupta et al. (2023); Wang & Gupta (2024)). Given the following data, the initially-learned rule is flies (X) :− bird(X) due to the data on "tweety". Then from the example "pengu" FOLD learns that penguins are an exception to the rule, so the rule flies (X) is refined by adding not ab(X).

```
% Data
bird(tweety).    flies(tweety).    bird(pengu).    penguin(pengu).    -flies(pengu).
% Learned Rules
flies(X) :- bird(X), not ab(X).                    ab(X) :- penguin(X).
```

FOLD has been used to encode models learned from various real-world data into s(CASP) programs. All versions of the FOLD algorithm inductively learn rule-sets from datasets, provide a comparable accuracy to deep learning systems (particularly, the FOLD-SE system), and are completely explainable, thanks to the s(CASP)'s justification framework. The example above is a simple

one for illustration purposes only, for more complete examples and formal review of using FOLD for ILP, please see Gupta et al. (2023) and Wang & Gupta (2024)

Discussion: Regarding induction, the paper by Lenat and Marcus state that "[. . .] by performing one step of reasoning, Cyc could generate tens of billions of new conclusions that follow from what it already knows". This has the possibility to add many new, default-true statements to the knowledge base. This creates inductive conclusions on a large scale, leading to increased generalizability. However, learning based only on internal knowledge without bringing in new knowledge could lead to increased bias in the conclusions drawn by some of Cyc's many reasoners due to normally imperceptible bias in the human-created axioms.

#04 Analogy: *Analogy is the ability to map similarity between objects of often disparate meaning. This is defined as a representational mapping from a known "source" domain to a novel "target" domain (Hall, 1989). In natural language, this commonly comes in the form of analogical phrases ("life is like a box of chocolates"). However, analogical reasoning is also used for various logic problems (such as Raven's Progressive Matrices). Analogical reasoning has been historically difficult for AI systems.*

s(CASP): Symbolic systems have been used to solve the analogical reasoning problem with various degrees of success (Mitchell, 2021). Because these other successful systems (Falkenhainer et al., 1989; Hofstadter et al., 1995) are often predicated on turning analogical problems into predicates that can then be reasoned over, s(CASP) could be used as a more modern reasoner with at least as much success. In addition, analogies can be automatically generated by taking a partially grounded s(CASP) rule, then "lifting" it by replacing constants with variables, then instantiating these variables with different constants. Thus, a concrete procedure for boiling water can be used to generate an identical procedure for boiling milk. Work is in progress to automate such analogical reasoning using s(CASP).
In the social chatbot application by Zeng et al. (2024), analogical reasoning has been realized by integrating s(CASP)'s reasoning with an LLM's ability to convert text into knowledge represented as logic predicates. If the LLM-based text-to-predicate translation module recognizes that the current topic is about the movie *Titanic*, a s(CASP)-based analogical reasoner is triggered to find, for example, *Catch Me If You Can* movie as a candidate for the next discussion topic, since *Leonardo DiCaprio* leads in both.

```
relevant_topic(A, B, 'has_same_person') :- movie(A), movie(B), person(C), act_in(C, A), act_in(C, B).
```

Additionally, LLMs can also help in generating analogical information, that is then used by the s(CASP) engine to gather analogous pieces of information:

```
relevant_topic('Batman Dark Knight Rises', 'Titanic', 'hero sacrifices himself for others').
```

Discussion: LLMs have been found to perform well at responding to natural language analogical phrases (Musker et al., 2024). However, in analogical reasoning, LLMs often make mistakes different than what humans would make due to a lack of deeper understanding. The ability to replicate responses to analogical queries is valuable but insufficient for true trustworthy AI. Alternatively, symbolic approaches can achieve good results on logical puzzles like RPMs, but lack the ability to integrate with natural language or make logical leaps like analogical phrases require without careful

data population. In that area, Cyc can do well only on analogical phrases contained in its knowledge base.

#05 Abductive Reasoning: *The term abduction refers to a form of reasoning that is concerned with the generation and evaluation of explanatory hypotheses. We could also think of abduction as assumptions-based reasoning. Abductive reasoning leads back from facts to a proposed explanation of those facts or assumptions that will explain that fact.*

s(CASP): More formally, abduction is a form of reasoning where, given the premise $P \Rightarrow Q$, and the observation Q, one surmises (abduces) that P holds. More generally, given a theory T, an observation O, and a set of abducibles A, then E is an explanation of O (where $E \subset A$) if:

$$T \cup E \models O \qquad\qquad T \cup E \text{ is consistent}$$

We can think of abducibles A as a set of assumptions. Generally, A consists of a set of propositions such that if $p \in A$, then there is no rule in theory T with p as its head (that is, there is no way to argue for p). We assume the theory T to be an answer set program. Under a goal-directed execution regime, an ASP system can be extended with abduction by simply adding the following rules for an abducible p:

```
p :- not neg_p.          neg_p :- not p.
```

This is automatically achieved for a predicate p that we want to declare as an abducible in the s(CASP) system through the declaration: #abducible p.

Discussion: The size of Cyc's knowledge base works in its favor for abductive learning. Since Cyc does not necessarily need to build proofs at runtime, but relies on a huge created knowledge base, it is simple to step "backwards" along a reasoning tree for a function to find a plausible explanation for why something is true or not.

#06 Theory of Mind: *Two of the desiderata deal purely with the way an AI system communicates with a user (this and #15). Theory of Mind involves building and continually updating nuanced models of conversation partners, including their knowledge, intentions, and communication styles, to interact appropriately. This helps the AI decide how terse or elaborate to be, when to ask clarifying questions, and how to adapt to ambiguity while avoiding miscommunication. Additionally, the AI must maintain a self-model to understand its capabilities, limitations, and role in the interaction.*

s(CASP): While Lenat and Marcus posed the combination of LLMs and Cyc as a potential solution for the shortcomings of both, that was purely an idea for the future. However, the neuro-symbolic approach of combining LLMs with s(CASP) has already been explored for a variety of domains (Zeng et al., 2023; Basu et al., 2021a). In this use case, the LLM provides a natural language semantic parser for a s(CASP) reasoner. This constrains the LLM to increase response truthfulness and fluency, and allows for easy interfacing with a symbolic system. Unlike other methods to improve LLM accuracy (such as chain-of-thought prompting or more intensive training), using a s(CASP) reasoner provides actual logically-grounded constraints. The s(CASP) system also determines the boundary of the "cognition" of AI with a dynamically maintained knowledge base, which contributes as an essential part to a self-model, i.e., to avoid hallucination and be aware that something beyond its capability exists. When the current dialogue is far beyond the task at

hand it will be labeled as "irrelevant" by the LLM-based semantic parser, driving the conversation back to the task. Additionally, attaching an LLM to s(CASP) allows s(CASP) to tailor its language according to the theory of mind as well as an LLM can.

Discussion: LLMs are very adaptable to the user, having been trained on numerous styles of text and speaking. Research finds that LLMs can automatically moderate their responses based on the tone (both spoken and written) of the user's input (Lin et al., 2024). However, unconstrained LLMs by themselves can be easily "jailbroken" to speak in inappropriate ways (Chao et al., 2023).

#07 Quantifier-fluency: *Quantifier fluency relates to being able to reason correctly with quantifiers. Quantifiers play a crucial role in interpreting meaning of sentences. For example, as Lenat & Marcus (2023) point out, the sentence "Every Swede has a king" versus the sentence "Every Swede has a mother" have to be interpreted appropriately.*

s(CASP): Since s(CASP) supports predicates, quantifiers can be easily modeled. In an answer set program, all variable are universally quantified. However, negation as failure can introduce universally quantified variables in the body of the rules. Consider for example, the following definition of a bachelor.

```
bachelor(X) :- man(X), not married_status(X).
married_status(X) :- married(X,Y,T). %X married Y at time T
```

The variables Y and T are existentially quantified in the body of the second rule. The negation of married_status in the first rule leads to these variables being universally quantified in the body. That is, X is a bachelor if X is a man, and did not marry *anyone* at *anytime*. Thus, goals with universal quantification can appear in the body of s(CASP) programs, allowing it to incorporate both types of quantifiers.
With respect to the example where we have to distinguish between every Swede having a king vs having a mother, such nuances can be modeled via ASP's global constraints, as illustrated below.

```
false :- swede(X), swede(Y), X ≠ Y, king(K1, X), king(K2, Y), K1 ≠ K2. % All Swedes have same king
false :- king_of_sweden(K1), king_of_sweden(K2), K1 ≠ K2. % There is only one King of Sweden
king(K,X) :- king_of_sweden(K), swede(X).

false :- swede(X), not hasmother(X). % Every Swede has a mother
hasmother(X) :- mother(X,Y).
```

Discussion: The way LLMs handle quantifier fluency is by leaning on prior knowledge to make educated guesses. However, it can wrongly fill in unknown information with unrelated assumptions. Cyc also relies on the completeness of its knowledge base for reasoning, and thus cannot reason without information that is not in or deducible/inducible from its knowledge base.

#08 Modal-fluency: *Modal-fluency is the ability to understand and use qualified statements. This includes statements such as "He hopes that it was successful" or "I believe she's afraid that it may cost her the job", which can be difficult for an AI system to reason over.*

s(CASP): ASP is based on stable model semantics that allow for multiple possible worlds. The semantics of modal logics that can be used to model modal-fluency is also given in terms of possible worlds. Thus, modal logics can be elegantly modeled with ASP/s(CASP). The $\neg K$

operator in epistemic modal logic, e.g., is nothing but default negation. Note that the concepts of "p is definitely true", "p may be true", "p is unknown", "p may be false", "p may be unknown", "p is definitely false" are represented by ASP expressions"p", "not –p", "not –p ∧ not p", "not p", and "–p", respectively, where –p represents strong negation (Gelfond & Kahl, 2014). Deontic logic, or the modal logic of obligations, can also be naturally represented in ASP/s(CASP). An obligation (the **OB** modal operator) can be represented as ASP's global constraint. In fact, since the odd loop over negation represents a global constraint, it can be used to elegantly resolve the contrary-to-duty paradox (as well as other paradoxes) of deontic logic. Consider the following statements for a housing community:

1. It ought to be the case that there are no dogs.
2. It ought to be the case that if there are no dogs, then there are no warning signs.
3. If there are dogs, then it ought to be the case that there are warning signs.
4. There are dogs.

In our framework, this is simply modeled as:

s(CASP): dog

```
dog :- not -dog, not dog.
:- -dog, not -warning_sign.
:- dog, not warning_sign.
dog.
```

Given this program, no world will be permitted in which there is no warning sign. If we remove (4), then no worlds will be possible in which a dog is present, as now constraint (1) will kick in. This ASP encoding thus elegantly resolves the contrary-to-duty paradox (a.k.a. the Chisholm paradox). For a more detailed discussion of the way s(CASP) allows for deontic logic representation, see Gupta et al. (2025)

Discussion: Although LLMs do not possess or store any knowledge separately, the amount of natural language they've internalized gives them some ability to "understand" complex sentences. Larger LLMs perform moderately well with modal logic, though there is still much room for improvement (Holliday et al., 2024). While the LLMs can parse modal sentences, they lack a true understanding of the logic behind modal statements. As with analogical reasoning (Desideratum #04), replicating response patterns to modal logic does not equate to an understanding of it, which makes them prone to strange mistakes and hallucinations.

#09 Defeasibility: *Defeasibility represents the ability to acquire new information and change previously held beliefs, many of which were likely only true by default in the first place. This requires the ability to reconsider previous beliefs and evaluate them on their own merits against new information before deciding whether the new or old information should be kept.*

s(CASP): Defeasibility can be implemented using default rules. Default rules are directly supported in ASP and s(CASP) with the help of negation-as-failure. Also, as discussed earlier, the FOLD algorithm primarily learns a model from data in the form of default rules with exceptions

(as discussed in Desideratum #03). These learned rules not only include exception, they may also include exception to an exception, exception an exception to an exception, and so on. This allows for a high level of fidelity when learning or representing a nuanced concept.
Note that default rules also allow ASP and s(CASP) to model human judgment. Consider a doctor prescribing a medicine M for disease D to patient P. Medicine M may have a side-effect for patient P, and so the doctor has to make a judgment call before prescribing. The doctor could aggressively jump to the default conclusion and may just ignore the possibility of a side-effect (contraindication).

```
prescribe(M, D, P) :- cures(M, D), not contraindicated(M, P).
contraindicated(M, P) :- has_side_effects(M, P).
```

Or, the doctor could be conservative in jumping to the default:

```
prescribe(M, D, P) :- cures(M, D), not contraindicated(M, P).
contraindicated(M, P) :- not -has_side_effects(M, P).
```

The rules above could be simplified and written, respectively, as:

```
prescribe(M, D, P) :- cures(M, D), not has_side_effects(M, P).      %aggressive
prescribe(M, D, P) :- cures(M, D), -has_side_effects(M, P).         %conservative
```

The first rule states that if there is no information about a possible side-effect, then proceed with prescribing. The second rule states that the possibility of a side-effect must be ruled out before prescribing. The ability to represent these nuanced judgment calls using default and strong negation is a positive aspect of ASP and s(CASP).
In the reliable chatbot mentioned in Desideratum #06, s(CASP) maintains the state of the conversation, tracking the progress of the task. Once new information comes, s(CASP) first checks if it is consistent with the current state and knowledge base, and then uses it to update the state. Inconsistent information will either be discarded or clarification will be sought from the human user.

Discussion: Both LLMs and Cyc have a similar approach to defeasibility, which is to yield to human input whenever given. For Cyc, this is a part of the continual hand-grooming of its large knowledge base. When information is added that conflicts with existing information, a human deconflicts the two either by removing one piece of information or adding additional context around it. Cyc also supports default rules. LLMs have a less permanent approach, wherein rather than removing data from its memory (a noted difficulty with LLMs (Blanco-Justicia et al., 2025)), the LLMs generates new responses when challenged. While human input is almost always correct relative to AI output, both systems do lack the ability to critically evaluate their answers themselves (Antoniou & Batsakis, 2023).

#10 Pro and Con Arguments: *Another facet of XAI represented in Lenat's desiderata is the "pro and con arguments" desideratum, which is a succinct description of the ability to explain why a decision may or may not be true. This covers the ambiguity of life, where things may not be objectively true but rather situationally true based on some confounding factors. Additionally, in any given situation there may be multiple answers that are correct enough.*

s(CASP): Answer set programs inherently generate pros and cons in the form of multiple worlds. When looking at a solution in one possible world, s(CASP) outlines which factors must be true or false to support that world. Additionally, if a proof of a goal g fails, one can immediately

query not g (which must succeed in s(CASP)) to understand the cause of failure (Murugesan et al., 2024). Similarly, *counterfactuals* can be computed in s(CASP) with ease (Dasgupta et al., 2025) to examine various what-if scenarios.

Discussion: This functionality is built into Cyc which provides arguments for and against each point it considers, along with heuristics for those arguments, at increasingly granular levels. LLMs, however, do not provide balanced pro and con arguments because they fold too easily against user disagreement (see Desideratum #09). Being unable to defend a premise when challenged makes it impossible to trust a fair evaluation of its own answers (Wang et al., 2023).

#11 Contexts: *Contextual understanding allows an AI to interpret and adapt knowledge, behavior, and communication based on situational, cultural, and temporal factors, such as knowing when cheering is appropriate or recognizing shifting truths over time. It must reason both within a context (e.g., belief systems or cultural norms) and about contexts (e.g., deciding whether an inference holds true in multiple settings).*

s(CASP): The multiple worlds reasoning of all ASP programs allows for the exploration of multiple contexts. Consider this example from Peter Norvig:

1. People can talk.
2. Non-human animals are not able to talk.
3. Human-like cartoon characters can talk.
4. Fish can swim.
5. A fish is a non-human animal.
6. Nemo is a human-like cartoon character.
7. Nemo is a fish.

The solution to the problem of whether Nemo can talk is difficult to represent in classical logic, but simple in ASP. In ASP there would be two mutually exclusive worlds, a "cartoon" world where Nemo can talk, and a "real" world where Nemo cannot. Of course, context can also be carried as an argument of a predicate in s(CASP).

Discussion: Cyc contains over 10,000 contexts that are nested within other contexts to delineate topics, time periods, real or not real, and more. This allows for a robust differentiation of contexts for answers that fit the situation. However, it is still susceptible to encountering contexts not present in its handmade knowledge base and slowdowns related to having to go down multiple different context chains to provide answers. Still, this is better than LLMs which have difficulty keeping context even from a single conversation if the conversation grows too large (Hatalis et al., 2024).

#12 Meta-Knowledge and Meta-Reasoning: *Meta-knowledge and meta-reasoning enable an AI to reflect on its own knowledge, assess the reliability and source of its beliefs, and understand its strengths and limitations in performing tasks. This includes the ability to introspect, to adapt strategies mid-process, to account for changes in reasoning over time, and to critically evaluate the reliability of its sources.*

s(CASP): Negation-as-failure gives us a form of meta reasoning because it allows us to reason about a failed proof, enabling correction. The co-inductive hypothesis set (CHS) maintained in s(CASP) (Arias et al., 2018) keeps track of everything that has been found to be true, further allowing it to solve queries about the proof process itself. Did we solve this through assumption (co-inductive) or was the reasoning well-founded (inductive)? Meta-programming and meta-reasoning

can also be explicitly incorporated in s(CASP) by making the CHS accessible to the knowledge engineer through appropriate built-ins.

Discussion: This function is available in a limited capacity in both LLMs and Cyc. LLMs can be asked whether their answer is correct or not, but are prone to over-correcting if any user feedback is given. Cyc has reasoners about its reasoning, and thus can perform meta-reasoning.

#13 Explicitly Ethical: *Trustworthy AI must adhere to a transparent ethical framework, guided by strong core principles. Crucially, the AI's ethical commitments should be visible and understandable to those it affects, forming an unchanging part of its foundational contract with users.*

s(CASP): It is possible for users to add constraints to a s(CASP) program, forcing it to be explicitly ethical. This is unlike deep learning systems which are difficult to correct or add constraints that are not in the data they were trained on. Vašíček et al. (2024) have used s(CASP) constraints to validate a patient controlled analgesia (PCA) pump modeled with event calculus, and Arias et al. (2024b) used s(CASP) in the automation of administrative processes. These two applications illustrate how ethical restrictions can be modeled as constraints. Additionally, recently Gupta et al. (2025) demonstrated that there is 1-1 mapping between modal logics and ASP, and how norms of obligation, impermissibility, and permission of deontic modal logic can be modeled elegantly in s(CASP) through *odd loop over negation*. Conditional norms as well as conflicting norms can be modeled. This expressiveness is crucial for representing ethical systems. Additionally, s(CASP) is fully open source, allowing for complete review of its source code.

Discussion: As LLMs are still often "jailbroken" to provide data against the terms of their use, it is clear that this principle is failed by LLMs. As for Cyc, without access to their systems it is hard to verify if it is explicitly ethical or not.

#14 Sufficient Speed: *An effective AI system must respond to a user query in a reasonable length of time. While in rare cases a long processing time may be acceptable, in most use cases users expect answers to their queries in a matter of seconds.*

s(CASP): Rather than spending a large time training, as with deep learning approaches, logic programming systems like s(CASP) perform their reasoning when the query is executed. Reasoning in real world environments with many objects, each having state and properties, becomes overwhelming and can result in long execution time for queries. To avoid this problem, many logic programs are kept small and domain specific. Thus these programs lack important context for reasoning.

The Virtually Embodied Common Sense Reasoning (VECSR) system is built on a foundation of s(CASP) reasoning to solve problems in simulated embodied environments (Tudor et al., 2025). The advantage of connecting a s(CASP)-based system to a virtual simulation (as with the other connections to LLMs and data mentioned above) is that there is ultimate access to the context required for decision making. However, this results in a large knowledge base and long durations for query-execution. The VECSR system employs several compile-time static analysis techniques that can bring the reasoning time in even large programs to under a second much of the time, faster than even LLMs (proven in Tudor et al. (2025)). And unlike LLMs, logical constraints can be used to guarantee correctness and executability. Using a unified and multi-domain reasoner like s(CASP)

also eliminates the need for switching between multiple reasoners. Thus, s(CASP) can be used for real-world, real-time reasoning with a high degree of accuracy.

Discussion: Processing speed has long been a downside of symbolic systems, like Cyc, as they must have a large knowledge base. Cyc relies on a complex series of reasoners and caching stratagems to manage speed of response. However, in this area LLMs enjoy the benefits of their upfront training time. Modern LLMs, once trained, are able to produce responses faster than a human would in most cases. However, the training of an LLM takes several months and large quantities of computing power.

#15 Sufficiently Lingual and Embodied: *Sufficiently Lingual and Embodied refers to the ability of an AI system to communicate sufficiently for the task at hand. And while in some applications, the minimum necessary linguistics may not include natural language at all (such as in systems designed to produce numerical answers for math problems), in most use-cases some natural language is optimal for human understanding. Even when answers can be provided with little language support, it is important to be able to explain "why" an answer was given. In most cases, knowledge and reasoning are language-independent, but a trustworthy AI will nonetheless be able to communicate well.*

s(CASP): Commonsense reasoning systems, including s(CASP), represent knowledge as predicates. To communicate with humans, these predicates must be translated to and from natural language. Machine translation of languages has been shown to be done well with pattern matching. Thus, we employ LLMs to perform this task with high accuracy. The reliable chatbots referenced in Desideratum #06 illustrate this very well. These reliable chatbot applications (Zeng et al., 2023) usually target some specific tasks, and achieve the goal by collecting as much necessary information as they can from the human users under the guidance of s(CASP)'s knowledge base. Take the restaurant recommendation chatbot (Zeng et al. (2023) as an example. A customer's food preferences, budget, family-friendliness requirements, etc., must be known before a restaurant recommendation is given. This is encoded as part of s(CASP)'s knowledge. The chatbot uses an LLM to translate user dialogs to gather these requirements represented as predicates, then makes a recommendation using its encoded logic. It is also able to explain in natural language why it made a specific recommendation. By using an LLM as a semantic parser, the s(CASP)-based chatbot is able to have linguistic proficiency similar to a human.
Additionally, s(CASP) has been used in a fully embodied way by the VECSR system described in Desideratum #14. Using a s(CASP)-controlled agent in an simulation environment, s(CASP) can be used to reason over a wide variety of external stimuli and perform tasks in an embodied way. For complete details, see Tudor et al. (2025).

Discussion: As mentioned in Desideratum #06, while LLMs are very adaptable in their manner of speech they are also prone to being jail-broken. Additionally, because of the quantity of data LLMs have consumed, they often need to be fine-tuned to cover a specific domain. Cyc can cover a variety of domains and does have some functionality for natural language, but it is noted to be rudimentary. In our opinion, the best approach to making AI lingual is to leverage LLMs as translation devices for converting text into a logic-based formalism and vice versa.

#16 Broadly and Deeply Knowledgeable: *An effective AI must possess or access a broad and deep foundation of world knowledge, from commonsense to specialized domains, enabling it to interact meaningfully and contextually with people. While memorizing facts is less critical in the internet age, a trustworthy AI must skillfully retrieve, understand, and reason about information in real time, drawing on structured sources and web services. Importantly, it should interpret the meaning, trustworthiness, and implications of knowledge, using logical, analogical, and inferential reasoning as humans do.*

s(CASP): There are multiple ways to incorporate new information into a s(CASP) system. The FOLD algorithms can induce from multiple forms of data, including traditional datasets and even images (Padalkar et al., 2024). NeSyFOLD in particular demonstrates how the advantages of deep learning (such as robust image analysis) can be leveraged with s(CASP) to add explainability and transparent decision analysis to AI systems. The ability to learn from data is critical to build a system that is broadly and deeply knowledgeable, as there is too much information available in the connected modern age to ever encode by hand. Data can also be added from an LLM as in the reliable chatbot research by Zeng et al. (2024) or from simulated environments as in VECSR and reasoned over using the s(CASP) system.

Discussion: LLMs consume vast quantities of data, however there is a growing concern that LLMs will run out of data before achieving true reasoning (Villalobos et al., 2022). Because of the large quantity of data LLMs must be trained on, there can often be incorrect or low-quality data included, which is then reflected in wrong answers provided by the LLM. Cyc's solution to this has historically been the creation of its bespoke, human-groomed knowledge base, which trades man-hours for accuracy. However, it is not possible to encode the breadth of knowledge LLMs gain through training by hand in an amount of time that allows for it to be kept up to date with the changing world.

#17 Inconsistency Detection: *An effective AI system must be able to detect inconsistencies and enforce invariants, something that is easy for humans. The AI system, for example, should be able to tell that a person cannot sit and stand at the same time (inconsistency) or that an alive human must always breathe (invariant).*

s(CASP): Inconsistencies and invariants can be modeled quite simply as global constraints in ASP, and are supported in s(CASP). Examples above will be coded as:

```
1 false :- person(X), sit(X), stand(X).
```

These constraints may be used in sophisticated reasoning tasks, such as if we ask a person sitting on a chair to turn on a light switch at the other end of the room, then we know that the person must stand up first.

Discussion: The ability to specify inconsistencies has not been presented as a desideratum by Lenat & Marcus (2023). However, it is an extensively used feature in commonsense reasoning. Humans learn a vast number of such inconsistencies and invariants and use them in reasoning. LLMs do not have an explicit representation of inconsistencies, and the lack of enforcing consistency is a source of hallucinations. Cyc's reasoners have to have inconsistency detection embedded, and it is likely that the various reasoners incorporate it.

#18 Multiple Possible Worlds: *An effective AI system should reason about possible worlds that can simultaneously exist. Humans have the ability to simultaneously represent multiple possible worlds in their mind and reason over each separately. For example, the real world and the world of cartoons are two different worlds with parts that are common (fish can swim in both worlds, for example) and certain parts that are inconsistent (fish talk like humans in the cartoon world and not in the real world).*

s(CASP): In s(CASP), multiple possible worlds are easily represented through even loops over negation as discussed in the Abduction and Context desiderata. s(CASP) explicitly supports co-inductive reasoning (which relies on greatest fixpoint semantics), essential to representing multiple possible worlds.

Discussion: LLMs do not have an explicit notion of multiple possible worlds, and this again can be a source of hallucinations. In fact, machine learning systems are often not even aware of such nuances, and conflating of multiple worlds can be a source of inaccuracy and hallucination. Cyc supports contexts which can be thought of as different worlds, however, additional logic is needed to distinguish between different contexts and to indicate whether one context is a sub-context of another one. Lack of support for multiple possible worlds is a legacy of Russell's insistence on only relying on well-founded constructs (Gupta et al., 2022). ASP/s(CASP) break away from this dogma and support non-well-founded semantics, crucial to supporting commonsense reasoning. _____

4. Conclusion

Table 1 shows that the 16 (+2) desiderata that trustworthy AI systems must support could be supported by s(CASP) either directly or through recent applications. As future work, we believe that in order for s(CASP) to enable truly reliable AI it is necessary to improve its execution efficiency, applying techniques such as the dynamic consistency check by Arias et al. (2022b), and to build a large foundation of commonsense knowledge like Cyc already possesses. Brittleness is another issue in symbolic knowledge representation, i.e., KB construction is error prone, and a small change can drastically change the meaning of a KB. This is somewhat alleviated by providing explanations and program debugging/tracing tools created for s(CASP). However, what will further help are tools that can statically analyze, for example, if adding a constraint will make a KB inconsistent. Nevertheless, s(CASP) has already proven its usefulness in specific applications, such as the IRMA system by Forsante Corp of Finland for automating the Clozapine drug delivery guidelines. IRMA is the first logic-based system certified under the EU Medical Device Regulation (class 2b) (Forsante Oy, 2025). Thus s(CASP) is used commercially while still being open source and freely available for anyone looking to create trustworthy AI applications.

Acknowledgements

We are grateful to Vinay Chaudhri for bringing the paper by Lenat and Marcus to our attention.

References

Abe, H., Ando, R., Morishita, T., Ozeki, K., Mineshima, K., & Okada, M. (2025). Abductive reasoning with syllogistic forms in large language models. *Human and Artificial Rationalities.*

Advances in Cognition, Computation, and Consciousness (pp. 3–17). Springer.

Antoniou, G., & Batsakis, S. (2023). Defeasible reasoning with large language models-initial experiments and future directions. *RuleML+RR'23 workshop proceedings* (p. 7687). CEUR.

Arias, J., Carro, M., Chen, Z., & Gupta, G. (2020). Justifications for Goal-Directed Constraint Answer Set Programming. *TC of the 36th Int'l. Conference on Logic Programming* (pp. 59–72).

Arias, J., Carro, M., Chen, Z., & Gupta, G. (2022a). Modeling and Reasoning in Event Calculus using Goal-Directed Constraint Answer Set Programming. *Theory and Practice of Logic Programming*, *22*, 51–80.

Arias, J., Carro, M., & Gupta, G. (2022b). Towards Dynamic Consistency Checking in Goal-Directed Predicate Answer Set Programming. *24th Int'l. Symposium on Practical Aspects of Declarative Languages, PADL'22* (pp. 117–134). Springer.

Arias, J., Carro, M., Salazar, E., Marple, K., & Gupta, G. (2018). Constraint Answer Set Programming without Grounding. *Theory and Practice of Logic Programming*, *18*, 337–354.

Arias, J., Moreno-Rebato, M., Rodriguez-García, J. A., & Ossowski, S. (2024a). Automated legal reasoning with discretion to act using s(LAW). *Artificial Intelligence and Law*, *32*, 1141–1164.

Arias, J., Moreno-Rebato, M., Rodríguez-García, J. A., & Ossowski, S. (2024b). Towards value-awareness in administrative processes: an approach based on constraint answer set programming. *39th ACM/SIGAPP Symposium on Applied Computing, SAC'24* (pp. 770–778). ACM.

Arias, J., Törmä, S., Carro, M., & Gupta, G. (2022c). Building Information Modeling Using Constraint Logic Programming. *Theory and Practice of Logic Programming*, *22*, 723–738.

Basu, K., Varanasi, S., Shakerin, F., Arias, J., & Gupta, G. (2021a). Knowledge-driven Natural Language Understanding of English Text and its Applications. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35(14)*, 12554–12563.

Basu, K., Wang, H., Dominguez, N., Li, X., Li, F., Varanasi, S. C., & Gupta, G. (2021b). CASPR: A Commonsense Reasoning-based Conversational Socialbot. *4th Proceedings of Alexa Prize*.

Blanco-Justicia, A., Jebreel, N., Manzanares-Salor, B., Sánchez, D., Domingo-Ferrer, J., Collell, G., & Eeik Tan, K. (2025). Digital forgetting in large language models: A survey of unlearning methods. *Artificial Intelligence Review*, *58*, 90.

Brewka, G., Eiter, T., & Truszczynski, M. (2011). Answer set programming at a glance. *Commun. ACM*, *54*, 92–103.

Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., & Wong, E. (2023). Jailbreaking black box large language models in twenty queries.

Chen, Z., Marple, K., Salazar, E., Gupta, G., & Tamil, L. (2016). A physician advisory system for chronic heart failure management based on knowledge patterns. *Theory and Practice of Logic Programming*, *16*, 604–618.

Cheng, K., et al. (2024). Inductive or deductive? Rethinking the fundamental reasoning abilities of LLMs. *arXiv preprint arXiv:2408.00114*.

Cropper, A., & Dumancic, S. (2022). Inductive logic programming at 30: A new introduction. *J. Artif. Intell. Res.*, *74*, 765–850. From https://doi.org/10.1613/jair.1.13507.

Dasgupta, S., Shakerin, F., Arias, J., Salazar, E., & Gupta, G. (2025). C3G: causally constrained counterfactual generation. *Proc. PADL 2025* (pp. 215–232). Springer.

Falkenhainer, B., Forbus, K. D., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, *41*, 1–63.

Forsante Oy (2024). IRMA: A logic-based ai decision module for managing chronic diseases. From https://forsante.com/en/for-technology-companies/.

Forsante Oy (2025). MDR Certificate to Forsante, Ltd. From https://shorturl.at/uwGMo.

Gebser, M., Kaminski, R., Kaufmann, B., & Schaub, T. (2014). Clingo= ASP+ control: Preliminary report. *arXiv preprint arXiv:1405.3694*.

Gelfond, M., & Kahl, Y. (2014). *Knowledge representation, reasoning, and the design of intelligent agents: Answer set programming approach*. Cambridge Univ. Press.

Gelfond, M., & Lifschitz, V. (1988). The Stable Model Semantics for Logic Programming. *5th International Conference on Logic Programming* (pp. 1070–1080).

Gupta, G., Rajashekaran, A., Tudor, A., Salazar, E., & Arias, J. (2025). Modeling deontic modal logic in answer set programming. https://arxiv.org/abs/2507.05519.

Gupta, G., Salazar, E., Varanasi, S. C., Basu, K., Arias, J., Shakerin, F., Li, F., & Wang, H. (2022). Tutorial: Automating Commonsense Reasoning. *Workshops of the 38th ICLP'23*. CEUR-WS.org. From https://ceur-ws.org/Vol-3193/paper2GDE.pdf.

Gupta, G., Wang, H., Basu, K., Shakerin, F., Salazar, E., Varanasi, S. C., Padalkar, P., & Dasgupta, S. (2023). Logic-based explainable and incremental machine learning. *Prolog: The Next 50 Years*, (p. 346–358).

Hall, B., et al. (2021). Knowledge-Assisted Reasoning of Model-Augmented System Requirements with Event Calculus and Goal-Directed Answer Set Programming. *8th Workshop on Horn Clauses for Verification and Synthesis, HCVS@ETAPS'21* (pp. 79–90).

Hall, R. P. (1989). Computational approaches to analogical reasoning: A comparative analysis. *Artificial intelligence*, *39*, 39–120.

Hatalis, K., Christou, D., Myers, J., Jones, S., Lambert, K., Amos-Binks, A., Dannenhauer, Z., & Dannenhauer, D. (2024). Memory matters: The need to improve long-term memory in LLM-Agents. *Proceedings of the AAAI Symposium Series*, *2*, 277–280.

Hofstadter, D. R., Mitchell, M., et al. (1995). The copycat project: A model of mental fluidity and analogy-making. *Advances in connectionist and neural computation theory*, *2*, 2–3.

Holliday, W. H., Mandelkern, M., & Zhang, C. E. (2024). Conditional and modal reasoning in large language models. *arXiv preprint arXiv:2401.17169*.

Holzbaur, C. (1995). *OFAI CLP(Q,R) Manual, Edition 1.3.3*. Technical Report TR-95-09, Austrian Research Institute for Artificial Intelligence, Vienna.

Lenat, D., & Marcus, G. (2023). Getting from generative AI to trustworthy AI: What llms might learn from Cyc. *arXiv preprint arXiv:2308.04445*.

Lenat, D. B. (1995). CYC: a large-scale investment in knowledge infrastructure. *Commun. ACM*, *38*, 33–38. From https://doi.org/10.1145/219717.219745.

Lin, G.-T., Chiang, C.-H., & Lee, H.-y. (2024). Advancing large language models to capture varied speaking styles and respond properly in spoken conversations. *arXiv preprint arXiv:2402.12786*.

Mitchell, M. (2021). Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, *1505*, 79–101.

Murugesan, A., Wong, I., Arias, J., Stroud, R., Varadarajan, S., Salazar, E., Gupta, G., Bloomfield, R., & Rushby, J. (2024). Automating Semantic Analysis of System Assurance Cases using Goal-directed ASP. *Theory and Practice of Logic Programming*, *24*, 805–824.

Musker, S., Duchnowski, A., Millière, R., & Pavlick, E. (2024). Semantic structure-mapping in llm and human analogical reasoning. *arXiv preprint arXiv:2406.13803*.

Padalkar, P., Wang, H., & Gupta, G. (2024). NeSyFOLD: A Framework for Interpretable Image Classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38(5)*, 4378–4387.

Rossi, F., & et al. (2025). AAAI 2025 presidential panel on the future of ai research. From https://aaai.org/about-aaai/presidential-panel-on-the-future-of-ai-research/.

Tudor, A. R., Arias, J., & Gupta, G. (2025). VECSR: Virtually Embodied Common Sense Reasoning System. From https://arxiv.org/abs/2505.02144.

Vašíček, O., Arias, J., Fiedor, J., Gupta, G., Hall, B., Křena, B., Larson, B., Varanasi, S. C., & Vojnar, T. (2024). Early Validation of High-level System Requirements with Event Calculus and Answer Set Programming. *Theory and Practice of Logic Programming*, *24*, 844–862.

Villalobos, P., Ho, A., Sevilla, J., Besiroglu, T., Heim, L., & Hobbhahn, M. (2022). Will we run out of data? Limits of LLM scaling based on human-generated data. *arXiv preprint arXiv:2211.04325*.

Wang, B., Yue, X., & Sun, H. (2023). Can ChatGPT defend its belief in truth? evaluating LLM reasoning via debate. *arXiv preprint arXiv:2305.13160*.

Wang, H., & Gupta, G. (2024). Fold-se: an efficient rule-based machine learning algorithm with scalable explainability. *International Symposium on Practical Aspects of Declarative Languages* (pp. 37–53). Springer.

Yu, D., Yang, B., Liu, D., Wang, H., & Pan, S. (2023). A survey on neural-symbolic learning systems. *Neural Networks*, *166*, 105–126.

Zeng, Y., Rajasekharan, A., Basu, K., Wang, H., Arias, J., & Gupta, G. (2024). A reliable commonsense reasoning socialbot built using LLMs and goal-directed ASP. *Theory and Practice of Logic Programming*, *24*, 606–627.

Zeng, Y., Rajasekharan, A., Padalkar, P., Basu, K., Arias, J., & Gupta, G. (2023). Automated interactive domain-specific conversational agents that understand human dialogs. *Practical Aspects of Declarative Languages* (pp. 204–222). Cham: Springer Nature Switzerland.