Reliable and Explainable Reasoning with Symbolic Modules in Agentic LLMs

David Houri

CentraleSupélec & QUT

Research Discussion – 27 June 2025

Motivation: Strengths and Limitations of LLMs

- LLMs show remarkable language understanding and generation.
- But they fail at:
  - Structured reasoning
  - Rule compliance (e.g., regulation, eligibility)
  - Arithmetic consistency
  - Hallucination-prone responses

  In many real-world contexts, **correct reasoning is critical**:
  - **Medical diagnosis:** dosage decisions, drug contraindications
  - **Legal reasoning:** eligibility, penalties, contractual rules
  - **Transportation:** airline baggage fees, visa policies
- **Problem:** Can we make LLM reasoning reliable and explainable?

Example

Prompt

Can you provide the mathematical proof that the sum of two odd numbers always results in an odd number?

LLM-Generated Proof (Llama2)

Let $x = 2a + 1$ and $y = 2b + 1$, two odd numbers.

$$x + y = (2a + 1) + (2b + 1) = 2(a + b + 1)$$

Since $2(a + b + 1)$ is even, *the sum is odd*.

LLMs are stochastic parrots — correctness is not guaranteed.

Solution 1: Chain of Thought Prompting

- Encourages the LLM to reason step by step
- Decomposes complex questions into simpler subproblems
- Effective for arithmetic, logic puzzles, or procedural tasks
- **Limitations:**
  - Reasoning remains heuristic and pattern-based
  - No guarantee of correctness

Solution 2: Agentic AI and Tool Use

- LLM delegates sub-tasks to external tools (Calculators, Search engines, Code interpreters) :
- Enables structured workflows and modular reasoning
- **Limitations:**
  - Agents are often task-specific and work best on constrained, algorithmic problems
  - Performance drops in tasks requiring deep contextual understanding or nuanced interpretation (legal contracts, medical records, ethical dilemmas)

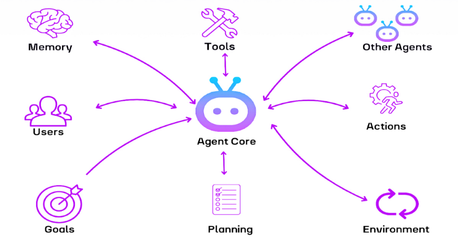


Figure: Possible Architecture of an agent-based model

Example Failure Case with CoT and Tools

Prompt

John is 63 years old and lives in France. According to the new tax regulation, retirees above 64 are exempt from this contribution. Is John exempt?

LLM Response

John is a retiree and the regulation says retirees above 64 are exempt. John is 63, so he qualifies for the exemption.

$\Rightarrow$ There is a need for mechanisms to **ensure the reliability and correctness** of model outputs.

Symbolic AI vs Neural Networks

Symbolic AI

- Based on explicit rules and logic (IF–THEN).
- Precise, interpretable, explainable reasoning.
- Good at following formal rules and constraints.
- Limited in handling ambiguity, uncertainty, and raw data.
- Difficult to scale and adapt to new domains.

Neural Networks (NNs)

- Learn patterns from large amounts of data.
- Flexible and powerful for perception and language.
- Handle ambiguity and noisy inputs well.
- Reasoning is implicit and often opaque (black box).
- Can produce errors and hallucinations.

Symbolic Agent

- Receives a query q and relevant facts $\mathcal{F}_q$.
- Applies a set of inference rules $\Phi = \{\varphi_1, ..., \varphi_n\}$ expressed as logical implications:

$$\varphi_i : \quad \text{IF } (A, r_1, B) \wedge (B, r_2, C) \text{ THEN } (A, r, C).$$

- **Derive conclusions** from $\mathcal{F}_q$ by applying these symbolic rules.

$\Rightarrow$ Produces logically grounded conclusions that can be used by the LLM to generate reliable and explainable answers.

Example of Symbolic Reasoning

Facts

```
isA(socrates, human).
isMortal(human, true).
```

Rule

```
isMortal(X, true) :- isA(X, human), isMortal(X, true).
```

Conclusion

```
isMortal(socrates, true).
```

Explanation

The symbolic agent uses the rule and facts to deduce that Socrates is mortal. This reasoning is explicit, verifiable, and explainable.

Towards Neuro-Symbolic AI

- Goal: combine the **rigor and explainability** of symbolic AI
- With the **learning power and flexibility** of neural networks
- Neuro-symbolic AI aims to build AI systems that:
  - Understand and manipulate explicit knowledge
  - Learn from raw data and generalize
  - Provide reliable and interpretable reasoning

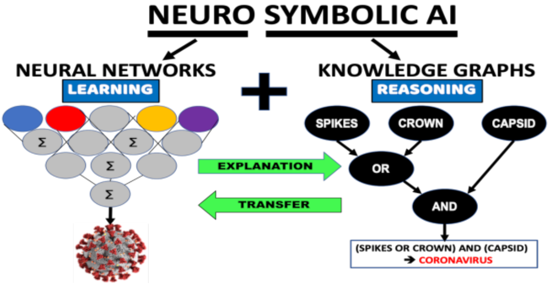
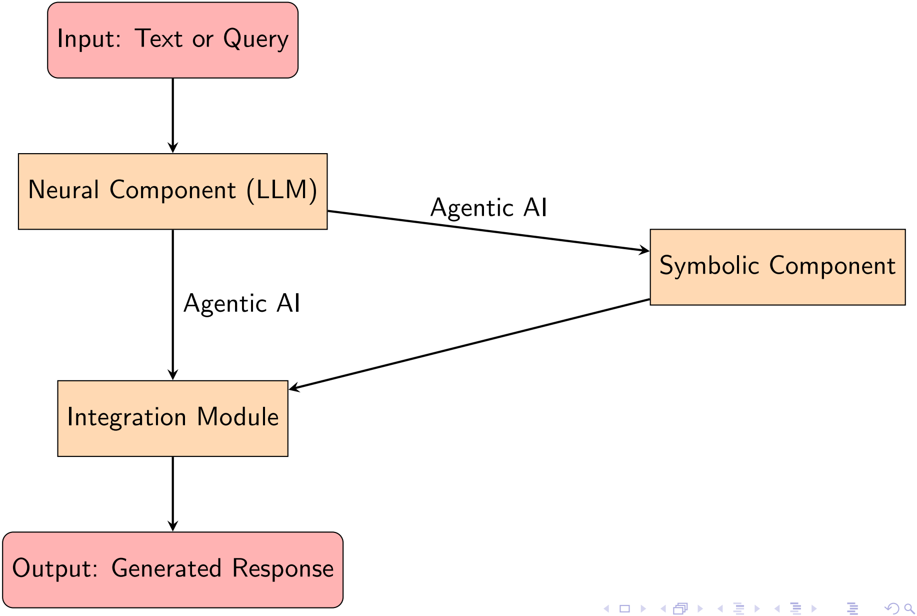


Figure: Illustration of a neuro-symbolic architecture

Proposed Architecture: Symbolic-Agentic Hybrid



Key Challenge: Extracting Rules and Facts from Context

- **Extracting symbolic rules and relevant facts** from unstructured or semi-structured text is a major challenge.
- This leverages GPT-4's strong language understanding but:
  - May lack formal guarantees on extraction quality.
  - Depends on prompt design and context size.
- Alternative or complementary approaches include:
  - **LesR module** (He et al.): specialized symbolic rule extraction methods.
  - **Fine-tuning language models** specifically for rule and fact extraction tasks.

Rule Extraction from Context

- **Extracting symbolic rules and relevant facts** from unstructured or semi-structured text is a major challenge.
- Current approach: **Use GPT-4 as a Retriever** to parse the context and extract logical rules and facts *before* answering.
- Alternative or complementary approaches include:
  - **LesR module** (He et al.): specialized symbolic rule extraction methods.
  - **Fine-tuning models** specifically for rule and fact extraction tasks.

Context Text:
"If a person is over 64 and retired, they are exempt from tax."

$\Longrightarrow$

```
exempt(X,tax) :- has(X,
age_gt_64), is(X,retired).
```

Test Case: Medical Diagnosis Prompt

User Prompt

"I've been experiencing persistent coughing for more than a week now, accompanied by a fever above 38°C. I also have a sore throat, fatigue, and nasal congestion. Recently, I've been feeling short of breath and experiencing chest pain when I breathe. I've also noticed some confusion and dizziness. I have a history of allergies and asthma, but I'm not sure if these symptoms are related to that."

Extracted Facts (Symbolic Representation)

- `has(patient, persistent_cough).`
- `has(patient, fever).`
- `has(patient, sore_throat).`
- `has(patient, fatigue).`
- ...

Symbolic Reasoning Output

Rules extracted from the pdf

```
diagnose(X, lower_resp_infection) :-
  has(X, persistent_cough),
  has(X, fever)

diagnose(X, upper_resp_infection) :-
  has(X, sore_throat),
  has(X, fatigue),
  has(X, nasal_congestion).

diagnose(X, pneumonia) :-
   ...,
   ...,
```

System Conclusion (GPT-4 + Symbolic Agent)

The patient may have a lower respiratory tract infection (cough + fever), and an upper respiratory tract infection (sore throat, fatigue, nasal congestion). No strong evidence for pneumonia or asthma. Conclusions are based on explicit rules and first-order reasoning, translated into natural language by the LLM.

Test Case: PTO Leave Policy Prompt

User Prompt

"Hi, I wanted to check how much time off I can take this year. I've been working full-time for 1 year and a half, and I'm based in the U.S. I haven't taken any vacation or sick days yet this year, and I don't have any leftover sick time from last year. Could you let me know how many days off I'm entitled to in total?"

Extracted Facts (Symbolic Representation)

- `is(user, full_time_employee).`
- `works(user, hours_per_week_ge_30).`
- `employment_length(user, months_gt_12).`
- `employment_length(user, months_le_60).`
- `pto_request(user, submit_at_least_72_hours_prior).`
- `pto_carry_over(user, hours_40_max).`

Symbolic Reasoning Output (PTO Policy)

Extracted Rules (from Document)

```
eligible(X, pto) :-
  is(X, full_time_employee),
  works(X, hours_per_week_ge_30).

pto_accrual(X, hours_9_99_per_month) :-
  employment_length(X, months_gt_12),
  employment_length(X, months_le_60).

pto_carry_over(X, hours_40_max) :- is(X, employee).
```

System Conclusion (GPT-4 + Symbolic Agent)

You are eligible for PTO as a full-time employee working over 30h/week. You accrue 9.99 hours of PTO per month, based on your 18-month employment length. You can carry over up to 40 hours of unused PTO. This answer is grounded in symbolic rules and facts extracted from your profile and the company policy PDF.

Benchmarks for Evaluating Reasoning

- **Tower of Hanoi (Benchmark 1)**
  (Shojaee et al., "The Illusion of Thinking", Apple, 2023)
  - Allows fast evaluation and clear metrics (success/failure). Best suited for Chain-of-Thought and agentic LLMs — less effective with a symbolic reasoning module.

- **RuleArena (Benchmark 2)**
  (Zhou et al., UCSB, 2024)
  - A benchmark designed for **rule-guided reasoning** in realistic, text-based scenarios.
  - Contains explicit rules for LLMs to follow.
  - Perfectly aligned with my approach:

Current Challenges in Implementation

- **Rule Extraction Instability**:
  - GPT-4 sometimes misses or misformats rules from context.
  - Inconsistent syntax or overly generic abstractions.
- **Symbolic Engine Limitations**:
  - Difficulty handling variables correctly across facts and rules.
  - Cannot manage uncertainty, negation, or defaults well.
- **Contextual Inconsistency**:
  - Entity linking and fact grounding are fragile.
  - Hard to maintain coherence across long or ambiguous user inputs.

Appendix 1: Prompt for Fact Extraction

```
You are a symbolic reasoning assistant that extracts logical facts from a case description.

Your goal is to extract **as many relevant and accurate facts as possible**, while ensuring they

{rules}

Each fact must follow this exact format (no period at the end):
predicate(entity, value)

Where:
- 'entity' is a lowercase constant (e.g., subject, user, person, item)
- 'predicate' must be chosen from those found in the rules (e.g., has, is, may_have, require, ca
- 'value' must use the same terms or categories that appear in the rules. Do **not** invent new

INSTRUCTIONS:
- Be thorough: extract **all relevant facts** that clearly appear or are directly implied by the
- Be consistent: reuse the **same predicates and value terms** as in the rules.
- Do **not** guess or invent values or predicates not found in the rules.
- Do **not** use uppercase letters for entities.
- Do **not** use any bullets, numbering, or extra explanations.
- Do not extract the conclusions from the rules document but the predicates.
```

Appendix 1: Prompt for Fact Extraction

```
NUMERIC VALUES:
If the description includes numerical values (e.g., "temperature of 39", "score less than 5"), a
- ‘_gt_N‘ for "greater than N"
- ‘_lt_N‘ for "less than N"
- ‘_ge_N‘ for "greater or equal to N"
- ‘_le_N‘ for "less or equal to N"
Use only thresholds that appear in the rules.

 Correct examples:
has(user, fever_gt_38)
has(subject, experience_lt_5)
is(candidate, eligible)
has(car, pressure_lt_90)
is(machine, active)

Now extract all relevant logical facts from the following case description :

\"\"\"{user_description}\"\"\"
```

Appendix 2: Prompt for Rule Extraction

Prompt

```
You are a symbolic reasoning engine.

Your task is to read a descriptive document and extract logical rules that can be used for auto

Each rule must follow **strictly** this format (without any period at the end):
head :- condition1, condition2, ..., conditionN

Where:
- ‘head‘ is the conclusion or result (e.g., eligible(X, refund))
- The conditions describe required facts using predicates (e.g., has(X, delay_gt_60), paid(X, s
- Use predicates like: has, is, may_have, require, triggers, includes, causes, eligible, locate
- Always use ‘X‘ as the main entity (e.g., traveler, person, document)
- Do **not** use natural language or explanations
- Do **not** add periods at the end of rules

 Pay special attention to the following:
1. Identify all **\if... then..."** patterns (even if implicit)
2. Identify all **numerical thresholds or conditions** (delays in minutes, percentages, payment
   - ‘_gt_N‘ for greater than N
   - ‘_lt_N‘ for less than N
   - ‘_ge_N‘ for greater or equal to N
   - ‘_le_N‘ for less or equal to N
   - ‘_eq_N‘ for equal to N (if applicable)
3. Encode all constants using lowercase and underscores only (e.g., more_than_one_hour → delay_g

 Correct examples:
eligible(X, refund) :- has(X, delay_gt_60), paid(X, single_payment), assembled_by(X, sncf_serv
```