Knowledge-Based Systems 351 (2026) 116657

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

GOFAI meets generative AI: Development of expert systems by means of large language models

Eduardo C. Garrido-Merchán [a,b], Cristina Puente [c,*]

[a] Quantitative Methods Department, DIGNUM Research Center, ICADE, Comillas Pontifical University
[b] Institute of Research in Technology (IIT), Madrid, Spain
[c] Computer Science Department, ICAI School of Engineering, Comillas Pontifical University, 28015 Madrid, Spain

ARTICLE INFO

Dataset link: https://github.com/eduardogarrido90/GOFAIGenAI

Keywords:
Expert systems
Language models
LLMs
Fact-Checking
hallucinations

ABSTRACT

The development of large language models (LLMs) has transformed knowledge-based systems such as open-domain question answering, which can automatically produce vast amounts of seemingly coherent information. Yet these models suffer from well-known limitations, most notably hallucinations, the confident generation of incorrect or unverifiable facts. In this paper, we introduce an approach to the development of expert systems that uses LLMs in a controlled and transparent way. By restricting the domain and employing a structured, prompt-based extraction protocol, we distil the knowledge of an LLM into a symbolic Prolog representation that can be inspected, validated and corrected by human experts, and queried by a deterministic inference engine. We evaluate the approach along three complementary axes. First, we assess factual accuracy through a manual expert audit and, to address the limited scale of manual checking, a large-scale automated cross-validation against Wikidata over three different LLM families (Claude Sonnet 3.7, GPT-4.1 and Grok 3), reporting Wilson confidence intervals and explicitly separating entity-linking error from factual error by means of a type-aware entity linker. Second, we show that the generated knowledge bases support genuine multi-hop deductive reasoning, negation as failure and aggregation when executed in SWI-Prolog, confirming that they behave as functioning expert systems rather than inert fact lists. Third, we analyze the statistical guarantees of the pipeline, making explicit how the autoregressive, non-independent nature of LLM generation affects the concentration bounds and confidence intervals. The result is a transparent hybrid solution that combines the recall of LLMs with the precision and explainability of symbolic systems, laying the foundation for dependable AI applications in sensitive domains. All code, prompts and generated knowledge bases will be made publicly available at https://github.com/eduardogarrido90/GOFAIGenAI upon acceptance.

1. Introduction

Since the first expert systems were developed in the 1970s and 1980s, such as DENDRAL [1,2], MYCIN [3] and XCON [4], this technique has not stopped evolving as an aid to knowledge and advice in a multitude of tasks across the medical, legal and educational domains, among others. The deployment of automated reasoning in such high-stakes domains, where outputs must be auditable and verifiable, remains an active concern [5].

One of the greatest complexities when creating an expert system is to provide it with complete and reliable knowledge. For this task, a human expert knowledgeable in the subject and numerous questionnaires or other methods were generally used to capture the knowledge obtained in graphs, decision trees, knowledge rules [6] in order to squeeze the knowledge obtained in the best possible way [7].

With the appearance around 2010 of large language models (LLMs), systems with an enormous capacity for storing data, a whole world of possibilities opened up for generating systems restricted to a smaller, task-specific domain. Chatbots for medicine, education and text completion were created in no time, allowing fluid dialogues and a multitude of new questions to be answered [8]. The problem is that not all the knowledge returned by the system is reliable. As indicated by [9,10] many times the LLM in its eagerness to give an answer produces hallucinations, i.e. wrong or not accurate knowledge as an answer. The problem here is serious, because if the sources of knowledge are not reliable, the answers should not be reliable either, so, unless we check it, the system will be spreading wrong information.

Hallucinations within an LLM are false responses that the system produces when there is either inaccurate data or an internal failure.

* Corresponding author.
E-mail address: cristina.puente@icai.comillas.edu (C. Puente).

https://doi.org/10.1016/j.knosys.2026.116657
Received 19 May 2026; Received in revised form 8 July 2026; Accepted 12 July 2026
Available online 21 July 2026

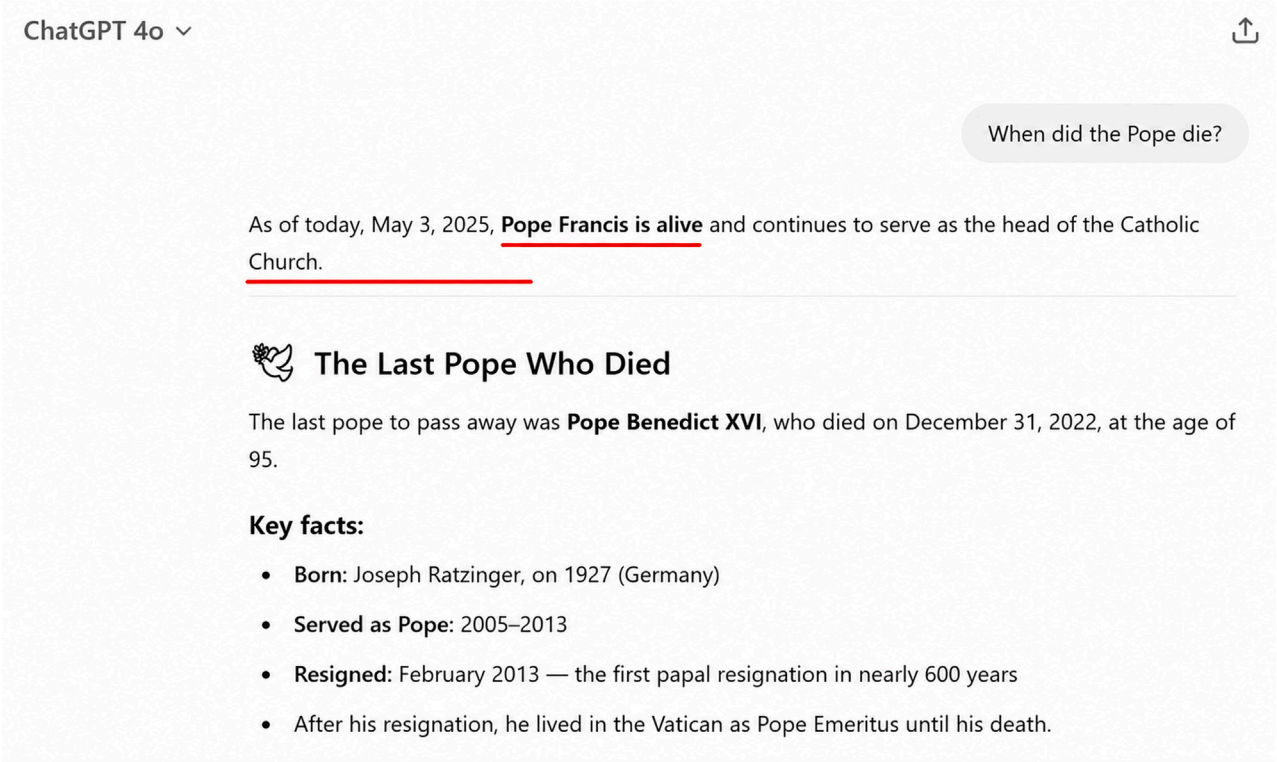

Fig. 1. Answer of ChatGPT 4o assuring that Pope Francis was alive on the 3rd of May 2025, when in fact he died on the 21st of April 2025.

Detecting these hallucinations is critical when implementing an LLM in a system, since it can induce serious errors in sensitive processes, such as medical, scientific, business, etc., and can contribute to the dissemination of false information in the networks due to lack of verification or interpretation [11].

The generation of hallucinations can be categorized as a failure for any of these reasons [12,13]: Data failure [14], because they are not up to date, there is biased data or they are mixed with fake news, intellectual property problems when accessing certain information sources, etc. In the case proposed in Fig. 1, there is a clear case of outdated information, since the system did not have in its data the death of Pope Francis 15 days before the question was asked. Even so, it asserts that as of May 3, the pope is still alive. To solve this problem, it is advisable to use reliable sources of information, contrasted information (for example in the case of medical issues, the Mayo Clinic has extensive information on its website https://www.mayoclinic.org/es), and if the information is not accessible or there is not enough information to formulate an adequate answer, do not answer the question.

A second source is failure in the model and its training, including architectural limitations, difficulties in grasping the context and the complexity of a question, and overfitting of the training data.

Finally, the answer-generation step itself can fail, because LLMs often prioritize the fluency of the generated text over the accuracy of its content. A high sampling temperature increases randomness, whereas concentrating on the highest-probability tokens can degrade textual quality [15].

There are several proposals to solve these problems, including using reliable external sources, doing fact-checking within the application itself to check that it shows the same results (using synonyms, different temperatures, etc.) or training the systems to detect erroneous answers [16,17].

To propose another solution to this problem, we have designed a system that addresses a smaller and more specific knowledge domain. Through a set of predefined prompts [18–21], an LLM such as ChatGPT generates a more manageable knowledge base, which is then transformed into an easily verifiable model, namely a system of logical rules [22] that a human expert in the field can inspect. This verification step confers three important advantages on the resulting expert system. First, the system is explainable, since every answer it returns is backed by an explicit clause or a finite resolution proof. Second, the volume of available information grows, because eliciting knowledge in this way is considerably faster and cheaper than interviewing a human expert. Third, the information becomes trustworthy, since any error that the expert detects in the symbolic model is corrected immediately and, once corrected, remains corrected for every future query.

Regarding the creation of an expert system, Prolog is a well-known logic programming language based on first-order predicate logic and has long remained popular for the development of expert systems due to its declarative pattern matching syntax and inference mechanism. System knowledge is expressed by facts and rules, and the system deduces new knowledge using backward chaining and resolution. Famous applications are medical diagnosis systems (e.g., MYCIN), legal reasoning tools [23], and configuration of complex products [24,25]. For example, Prolog was the programming language employed by the clinical question answering system CLINIQA [26] and by XCON (also called R1), a rule-based system used at Digital Equipment Corporation to specify how to configure machines [27]. Its capacity of dealing with symbolic reasoning makes it an appropriate tool for constructing tutoring systems and intelligent advisors [28,29]. Prolog is still a very good option to program expert systems in the fields which require explainability, transparency and rule based reasoning.

The contribution of this work is not any single one of these ingredients in isolation, since prompt-based extraction, symbolic encoding and human validation are individually well known, but rather the closed, auditable loop that binds them together and the guarantees it affords. We make four specific contributions. First, we describe a recursive, ontology-aligned extraction protocol that distils an LLM into a first-order Prolog knowledge base under a controlled predicate vocabulary. Second, we develop a statistical verification methodology that turns expert spot-checking into a hypothesis test with explicit

sample-complexity bounds, and we make transparent how the non-independent, autoregressive nature of LLM generation affects those bounds and the associated confidence intervals. Third, we add an automated and reproducible cross-validation stage that grounds the extracted facts against Wikidata over three different LLM families and that, by means of a type-aware entity linker, separates entity-linking error from genuine factual error. Fourth, we show that the resulting knowledge bases are not inert fact lists but executable expert systems that support multi-hop deductive reasoning, negation as failure and aggregation. To the best of our knowledge, the integration of LLM knowledge distillation, symbolic Prolog encoding, statistical reliability guarantees and automated knowledge-graph cross-validation into a single human-in-the-loop pipeline has not been proposed before.

To achieve these goals, we have built a system that queries the information from the LLM through a set of specific prompts designed by an expert, then generates a model of logical rules to ease the expert's validation phase, together with a correction method to clean the information from imprecisions, wrong data or data bias, leading to the creation of a reliable expert system. So, the paper is divided as follows. First, we start by introducing how expert systems have evolved since the 1970s and the role that Large Language Models (LLMs) have played in a new era of knowledge extraction for systems such as chatbots and recommendation engines. We have remarked that LLMs tend to hallucinate, generate incorrect or unverifiable outputs, which can cause harmful decisions in important tasks, such as medicine, education, law.

Section preliminary and related works, presents a state of the art of the works that have served as basis for this project, reviewing its achievements and main contributions. In the methodology section, we have developed a protocol to select, define and process the knowledge of an LLM. For this task, we have used customized prompts to extract structured information from language models. This information is represented in Prolog, which excels in transparency, explainability and rule-based reasoning. In the methodology section we have presented a recursive procedure which constructs conceptual graphs interrogating the LLM, translating its response to Prolog facts and relations, and validating the knowledge syntactically and semantically. The experimental section shows both quantitative and qualitative results. We also demonstrate that the Prolog expert systems generated can be successfully executed and queried without errors, confirming their feasibility for practical use. Finally, in the last section, conclusions and future works are presented.

2. Preliminary and related work

In this section, we will review the work that has been developed in this area and has served as the basis for our system. We will divide them into three main parts. On the one hand, information gathering, where we will address the method used to query the LLM information. Secondly, we will focus on the formal representation of the knowledge and its validation, and finally we will address the construction of the expert system.

2.1. Knowledge acquisition

Since 2010, with the appearance of large language models such as LLaMA, GPT-3 and Copilot, applications that rely on these systems have not ceased to appear, ranging from recommendation systems and chatbots to classifiers and automatic programmers, and they have transformed the digital landscape.

Knowledge generation has undergone an algorithmic treatment different from traditional methods, such as fine-tuning [20] or knowledge extraction from the LLM itself through prompts [30]. The latter is typically used to reduce the broad knowledge of the LLM to a more specific, controllable and transparent domain. Structured extraction of semantic information from an LLM through carefully engineered prompts has been shown to be effective in adjacent settings, such as the extraction of semantic code information from software repositories [31]. Building on these works, we design a complete set of prompts that cover the knowledge domain we want to test in our experiments.

2.2. Neuro-symbolic knowledge extraction and knowledge graph construction

Our work sits within the broader programme of neuro-symbolic artificial intelligence, which seeks to combine the pattern-recognition strengths of sub-symbolic neural models with the transparency and rigour of symbolic reasoning [32]. Two lines of research are particularly relevant. On the one hand, language models have been studied as implicit repositories of relational knowledge that can be probed and read out as facts [33], and several methods construct knowledge graphs directly from the parameters of a pre-trained model [34]. On the other hand, the automated construction and, crucially, the fact-checking of knowledge graphs provide the representational and quality-assurance backdrop for such efforts [35]. Our contribution differs from these strands in three respects. First, we target a first-order logic-programming representation, Prolog, rather than a triple store, so that the extracted knowledge is directly executable by a deductive inference engine. Second, we wrap the extraction in an explicit human-in-the-loop validation protocol with statistical guarantees, rather than reporting raw probing accuracy. Third, we cross-validate the extracted facts against an external, openly editable knowledge graph, Wikidata, and we isolate the contribution of entity linking, a known bottleneck in knowledge graph construction [36–38], from the contribution of genuine factual error.

2.3. Limitations of LLMs regarding verification systems and computability

Even though large language models (LLMs) possess incredible abilities in solving everything from natural language generation to code completion, performance starkly demarcates boundaries put in place by classical computability theory. For example, when prompted to check if an arbitrary piece of code will terminate, LLMs will sometimes provide seemingly convincing rationale or even provide an answer authoritatively. This is not, however, a result of any decision-making property but rather statistical inference from training data. The Halting Problem, which Turing proved undecidable, states that there exists no algorithm — not even implicitly encoded ones in LLMs — capable of deciding termination for all possible programs [39]). The confident prediction by the model is therefore a simulation of understanding rather than actual verification. Analogously, when prompted to prove something in formal logic or do stepwise algebraic manipulations, LLMs often hallucinate intermediate steps or misapply logical principles, demonstrating their lack of formal basis.

It is this epistemic gap which has special resonance in areas where correctness is not a matter of plausibility, but proof, e.g. formal methods, verification of cryptographic protocols, theorem proving. The difficulty is not one of scale but principle: LLMs, bounded as they are by the Church–Turing thesis, work wholly within the universe of computable functions and therefore share the undecidability and incompleteness theorems which constrain all formal systems of adequately high complexity. What LLMs provide accordingly is a probabilistic shadow of mathematical argument good for heuristic exploration, but incapable in principle of delivering guarantees. In this manner, then, LLMs' illusion of omniscience may well obscure the very theoretical boundaries that are as relevant in the age of AI as in Gödel's and Turing's era.

3. Methodology

This section presents the methodological framework for extracting structured, symbolic knowledge from a large language model (LLM) and encoding it into a Prolog-based expert system. The objective is to construct a formal, logically consistent knowledge base representing the semantic landscape surrounding a user-defined concept. The proposed system combines recursive prompt chaining, ontology-aligned relation extraction, and symbolic encoding in first-order logic.

The pipeline is implemented in Python and invoked via a shell interface that receives a root keyword $T \in K$, where K denotes the set of all conceptual domains. The knowledge acquisition process is driven by two hyper parameters: the horizontal breadth $h \in N$, which limits the number of semantically related concepts retrieved per node, and the vertical depth $d \in N$, which defines the maximum depth of the conceptual graph rooted at T. The system interacts with the LLM via structured prompts, receiving a JSON-formatted response with concept nodes and labeled semantic relations, which are translated into Prolog facts using a controlled predicate vocabulary.

Formally, the output is a directed labeled graph $G = (C, R)$, where $C \subseteq K$ is the set of extracted concepts, $R \subseteq C \times L \times C$ is the set of relations among them, and L is a finite set of predicate labels comprising both generic forms (`related_to`, `implies`, `causes`, `relates_to`) and domain-specific extensions. To manage graph expansion, we define a recursive semantic expansion operator: $\Psi : C \longrightarrow \mathcal{P}(C \times L \times C)$ which maps a concept to a set of at most h labeled relations, and is recursively applied along newly discovered nodes up to depth d. The traversal is breadth-first and avoids cycles via memorization of visited concepts. We summarize these steps in Algorithm 1.

Algorithm 1: Recursive Semantic Expansion and Symbolic Encoding

Input: T: String // Root concept
h: Integer ≥ 1 // Horizontal breadth
d: Integer ≥ 0 // Vertical depth
Output: K_T: Prolog file // Prolog knowledge base

procedure BUILD_KNOWLEDGE_BASE(T, h, d)
  $Q \leftarrow [(T, 0)]$ ▷ Queue of (concept, depth)
  $V \leftarrow \emptyset$ ▷ Visited concepts
  $F \leftarrow \emptyset$ ▷ Set of unique fact hashes
  $KB \leftarrow \emptyset$ ▷ Prolog fact accumulator
  ;
  While $Q \neq \emptyset$
    $(c, \text{depth}) \leftarrow Q$.dequeue()
    If $c \in V$ **or** depth $> d$
      continue
    Endif
    $V \leftarrow V \cup \{c\}$
    Json $\leftarrow$ Query_LLM(c, h)
    (Concepts, Relations) $\leftarrow$ Parse_JSON(Json)
    ForAll $c' \in$ Concepts
      If $c' \notin V$
        Q.enqueue($(c', \text{depth} + 1)$)
      EndIf
      $KB \leftarrow KB \cup \{\text{concept}(c'), \text{related\_to}(c', c)\}$
    EndFor
    ForAll $(s, r, t, \text{explanation}) \in$ Relations
      $KB \leftarrow KB \cup \{r(s, t)\}$
      Store_Explanation($r(s, t)$, explanation)
    EndFor
    Deduplicate(KB, F)
  ;
  EndWhile
  Encode_KB(KB) $\rightarrow K_T$
  Validate_Prolog(K_T)
  return K_T
end procedure

Each extracted concept c is encoded as a unary predicate concept(c)., and each semantic relation $(c1, r, c2)$ as a binary predicate $r(c1, c2)$., where $r \in L$. Natural language explanations generated by the LLM are preserved as inline Prolog comments using % Explanation: directives. The system enforces uniqueness through lexical normalization and hashing of facts, and performs syntax validation using SWI-Prolog before deployment.

This algorithm supports the construction of multi-layered conceptual graphs rooted in an arbitrary input topic, enabling recursive reasoning and logical querying within an interpretable and extensible expert system framework.

The system supports both generic and domain-specialized relation extraction, enabling the construction of knowledge graphs that vary in granularity and scope. In its default configuration, the model adapts prompts based on the classified domain of the input topic — such as philosophy, history, or literature — yielding relations like developed_by, lived_in, wrote, pioneered, or invented, which capture biographical, geographical, and authorship information. This allows for richer, discipline-specific knowledge representations. However, the system can be configured to operate in a minimal mode, restricting the ontology to the core predicates concept/1, related_to/2, implies/2, and causes/2, thereby focusing purely on abstract conceptual structure and logical causality. This modularity enables flexible deployment for applications requiring either high interpretability or lightweight logical inference. We describe all the predicates in Tables 1 and 2.

As a final component, we developed a visualization tool that automatically parses the generated Prolog knowledge base and renders its conceptual structure as a directed graph. The visualizer distinguishes core causal predicates (e.g., causes/2, implies/2, related_to/2) from domain-specific relations (e.g., written_by/2, developed_by/2) using color-coded edges, and outputs as a labeled PDF graph. This facilitates rapid inspection of the knowledge topology, supports qualitative validation, and enhances interpretability across disciplinary contexts. We include examples in Figs. 2 and 3.

In both cases, Fig. 2 and Fig. 3, the user runs a shell script with all the configuration of the system that wants to generate, the shell scripts sends it to a Python knowledge extraction process that communicates through the LLM API with the LLMs. Then, the LLM sends back the knowledge required by the expert system builder to persist it in a Prolog file. Finally, the user can verify the information of the LLM in the expert system Prolog file. We summarize the diagram flow of all the logic presented in the methodology section in Fig. 4.

3.1. Theoretical framework: LLMs as probabilistic knowledge oracles

This section formalizes the theoretical foundations underlying the extraction of symbolic knowledge from large language models, establishing the connection between probabilistic language models and deterministic expert systems.

Let $\mathcal{L}$ be a large language model defining a probability distribution $P_{\mathcal{L}}(y|x)$ over responses y given a prompt x. We conceptualize the LLM as a *probabilistic knowledge oracle* that can be queried about factual relations. Formally, a large language model $\mathcal{L}$ induces a probabilistic knowledge oracle $\mathcal{O} : C \times \mathcal{R} \times C \rightarrow [0, 1]$, where C is the set of concepts and $\mathcal{R}$ is the set of relation types. The oracle is defined as:

$$\mathcal{O}(c_1, r, c_2) = P_{\mathcal{L}}(\text{“true”} \mid \pi(c_1, r, c_2)) \tag{1}$$

where $\pi(c_1, r, c_2)$ is a structured prompt querying whether the relation $r(c_1, c_2)$ holds. In practice, we do not query individual facts but rather request the LLM to generate all relations for a given concept, which can be seen as a batch query returning the set $\{(r, c_2) : \mathcal{O}(c_1, r, c_2) \geq \tau_{implicit}\}$ for some implicit threshold $\tau_{implicit}$ determined by the model's generation process.

The extraction process can be characterized as thresholded inference over this oracle. Given an oracle $\mathcal{O}$, a root concept c_0, expansion parameters (h, d), and implicit threshold τ, the extracted knowledge base is:

$$KB_{\tau}(c_0, h, d) = \bigcup_{c \in \text{Expand}(c_0, h, d)} \{r(c, c') : \mathcal{O}(c, r, c') \geq \tau\} \tag{2}$$

where $\text{Expand}(c_0, h, d)$ denotes the breadth-first expansion of concepts up to depth d with branching factor h.

Table 1
Core predicates supported by the system.

Category	Predicate	Arity	Description
Core (Causal)	concept/1	1	Declares a concept
	related_to/2	2	Links a concept to another
	implies/2	2	Logical implication between concepts
	causes/2	2	Causal relation between concepts
	relates_to/2	2	Generic semantic relation

Table 2
Domain-specific predicates supported by the system.

Category	Predicate	Arity	Description
Philosophy	response_to/2	2	Theory formulated as response to another
	school_of_thought/2	2	Philosopher belongs to school
	main_work/2	2	Main philosophical work of thinker
	lived_during/2	2	Philosopher lived during period
	developed_by/2	2	Concept developed by philosopher
	influenced_by/2	2	Philosopher/work influenced by another
	criticized_by/2	2	Theory criticized by philosopher
Literature	written_by/2	2	Work written by author
	published_in/2	2	Work published in year
	set_in/2	2	Setting or period of literary work
	protagonist_of/2	2	Character is protagonist of work
	genre_of/2	2	Genre of a work
	movement/2	2	Work or author belongs to movement
	influenced_by/2	2	Work influenced by another work
	adapted_into/2	2	Source work adapted into another form
Arts	created_by/2	2	Artwork created by artist
	created_in/2	2	Year or period of creation
	belongs_to/2	2	Artist/work belongs to movement/style
	housed_in/2	2	Artwork housed in location
	technique_used/2	2	Artistic technique applied
	commissioned_by/2	2	Work commissioned by patron
	trained_under/2	2	Artist trained under another
	influenced_by/2	2	Artistic influence from earlier artist
History	born_in/2	2	Person born in year/place
	died_in/2	2	Person died in year/place
	occurred_in/2	2	Event occurred in time period
	located_in/2	2	Entity located in place
	preceded/2	2	One event/entity precedes another
	succeeded/2	2	One event/entity succeeds another
	founded_by/2	2	Institution founded by person
	ruled_during/2	2	Ruler ruled during period
	contemporary_of/2	2	Temporal coexistence between people

This formalization allows us to derive precision bounds via concentration inequalities. Let p denote the true precision of the oracle, defined as the probability that a fact asserted by the LLM is correct with respect to ground truth. Given n independently sampled facts from the knowledge base, let $\hat{p}$ be the empirical precision. By Hoeffding's inequality, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$:

$$|p - \hat{p}| \leq \sqrt{\frac{\ln(2/\delta)}{2n}} \tag{3}$$

since each fact verification constitutes a Bernoulli trial with success probability p. In our experimental setup with $n = 250$ verified facts and observed precision $\hat{p} = 0.992$, with confidence $1 - \delta = 0.95$, we obtain $p \geq 0.992 - 0.086 = 0.906$, providing a lower bound on the true precision significantly above our null hypothesis threshold of 0.80.

The knowledge extraction process admits an information-theoretic interpretation as well. Let $\mathcal{D}$ represent the true state of knowledge about a domain, modeled as a random variable over possible world states. The information gain from extracting knowledge base KB about domain $\mathcal{D}$ is $I(KB; \mathcal{D}) = H(\mathcal{D}) - H(\mathcal{D}|KB)$, where $H(\cdot)$ denotes Shannon entropy. The goal of efficient knowledge extraction is to maximize $I(KB; \mathcal{D})$ while minimizing the number of queries to the LLM oracle. The recursive expansion strategy employed in our algorithm approximates a greedy maximization of information gain, as semantically related concepts are likely to reduce uncertainty about the domain more than unrelated ones.

This theoretical framework establishes that our approach rests on principled foundations connecting probabilistic language models to symbolic knowledge representation, rather than constituting merely an engineering solution.

3.2. Sample complexity and PAC guarantees

We establish formal guarantees on the sample complexity required to extract a knowledge base with bounded error, drawing on the Probably Approximately Correct (PAC) learning framework [40].

Consider a domain with a finite set of entities $\mathcal{E}$ with $|\mathcal{E}| = n$ and a set of binary predicates $\mathcal{P}$ with $|\mathcal{P}| = m$. The hypothesis space of possible knowledge bases is $\mathcal{H} = \{KB \subseteq \mathcal{P} \times \mathcal{E} \times \mathcal{E}\}$, with size bounded by $|\mathcal{H}| \leq 2^{m \cdot n^2}$ for binary relations. In practice, knowledge bases are sparse, and we consider the realizable case where the true knowledge base KB^* has at most k facts.

An extraction algorithm $\mathcal{A}$ is (ϵ, δ)-PAC if, for any true knowledge base KB^* and any distribution over queries, with probability at least $1 - \delta$ over the randomness of the LLM responses, the algorithm outputs $\widehat{KB}$ such that the symmetric difference error satisfies:

$$\text{err}(\widehat{KB}, KB^*) = \frac{|(\widehat{KB} \setminus KB^*) \cup (KB^* \setminus \widehat{KB})|}{|KB^*|} \leq \epsilon \tag{4}$$

Under this framework, let the LLM oracle have per-fact error probability $\eta < 0.5$. To achieve (ϵ, δ)-PAC extraction for a knowledge base

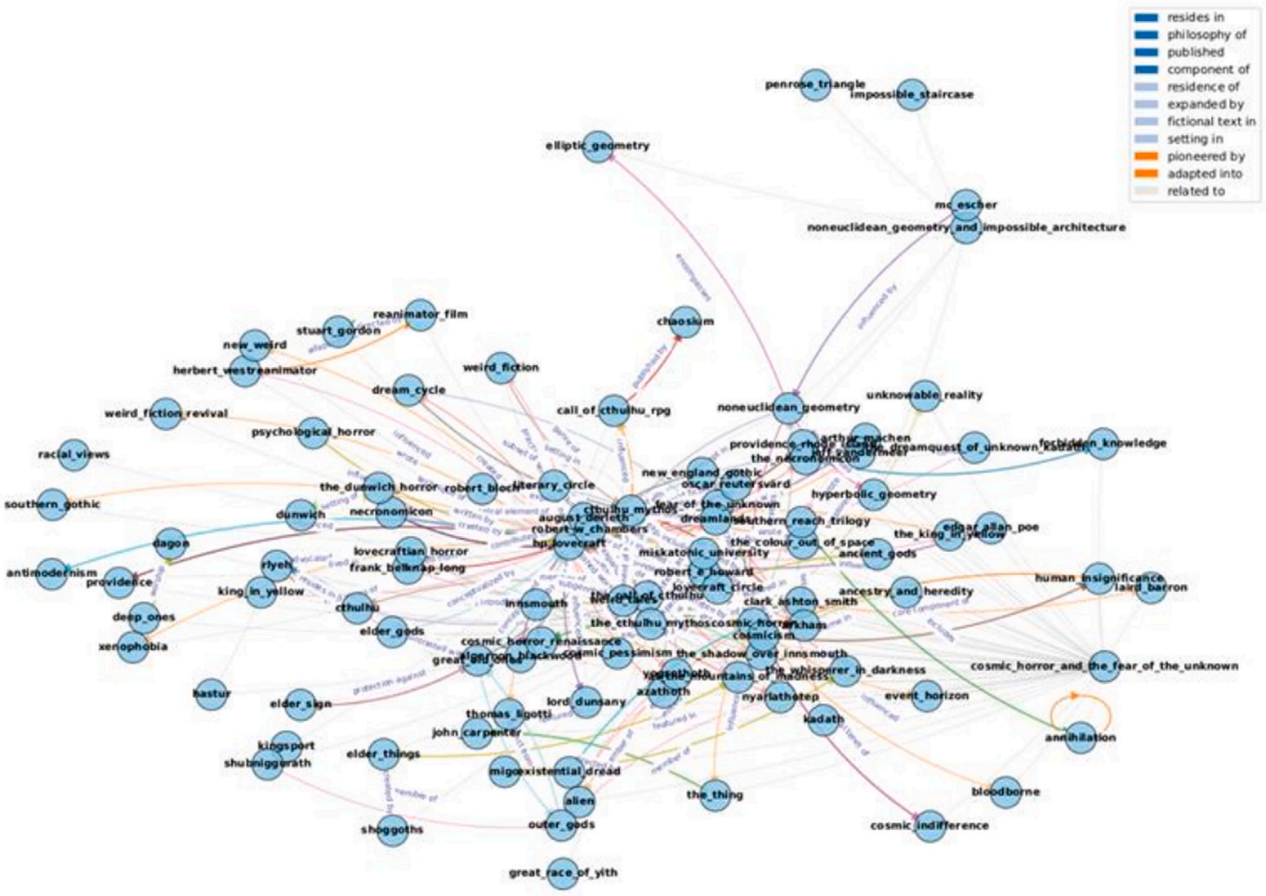

Fig. 2. Knowledge visualization of 100 nodes of causal and domain relations extracted about the cosmic horror author H. P. Lovecraft from Claude 3.7 Sonnet.

with at most k facts over n entities and m predicates, it suffices to verify:

$$N \geq \frac{1}{2\epsilon^2} \ln\left(\frac{2mn^2}{\delta}\right) \tag{5}$$

randomly sampled facts from the extracted knowledge base. This bound follows from applying a union bound over all possible facts in $\mathcal{P}\times\mathcal{E}\times\mathcal{E}$ combined with Hoeffding's inequality for each fact's verification, ensuring that the probability of any fact having empirical error exceeding ϵ from its true error is bounded by δ when N satisfies the stated condition.

For a typical domain in our experiments with approximately $n = 100$ entities (concepts extracted per topic), $m = 15$ predicates (as defined in Tables 1 and 2), target error $\epsilon = 0.05$, and confidence $1 - \delta = 0.95$, direct substitution yields $N \geq 200 \cdot \ln(6 \times 10^6) \approx 3120$. However, this represents a conservative upper bound. The effective hypothesis space is substantially smaller due to three factors: sparsity, since most entity pairs have no relation; domain constraints, as many predicate-entity combinations are semantically invalid; and correlation, given that facts are not independent and verifying one often implies others. Accounting for these factors with an effective hypothesis size of $|\mathcal{H}_{eff}| \approx 10^4$, corresponding to the empirically observed number of plausible facts, yields $N_{eff} \geq 200 \cdot \ln(4 \times 10^5) \approx 258$.

This theoretical requirement of approximately 258 verified facts aligns remarkably well with our experimental design of $n = 250$ manually verified assertions, providing theoretical justification for our evaluation methodology. The analysis yields several practical insights: our experimental verification of 250 facts is theoretically sufficient for the claimed precision bounds; for larger domains, the required sample size grows only logarithmically with domain size, ensuring scalability; and the framework allows practitioners to determine appropriate verification sample sizes for their target precision and confidence levels.

3.3. On the independence assumption and dependence-robust guarantees

The concentration results of Sections 3.1 and 3.2 are stated for independent verification trials, and this assumption deserves scrutiny. A large language model generates text autoregressively, so the facts it emits within a single knowledge base are not independent: a concept introduced early conditions everything generated afterwards, and an early hallucination can bias later assertions toward consistency with the initial error. Treating the correctness indicators of all extracted facts as independent Bernoulli trials would therefore overstate the effective sample size and produce confidence intervals that are too narrow. We make the dependence structure explicit and show that, once it is accounted for, the guarantees survive in a slightly weakened but still useful form.

Let the extracted facts be partitioned into G groups by root topic, where group g contains n_g facts and $n = \sum_{g=1}^{G} n_g$. Each group is produced by an independent invocation of the pipeline, with its own root prompt and its own sequence of API calls, so correctness indicators in different groups are independent. Within a group, the indicators $X_{g,1}, \ldots, X_{g,n_g} \in \{0,1\}$ may be arbitrarily dependent because they share a generative trajectory. Let p be the true accuracy and $\hat{p} = n^{-1} \sum_{g,i} X_{g,i}$ the pooled estimator. Writing ρ for the intra-group (intra-topic) correlation of the indicators, the variance of $\hat{p}$ is inflated relative to the independent case by the design effect

$$\mathrm{Deff} = 1 + (\bar{n} - 1)\,\rho, \qquad \bar{n} = \frac{1}{G}\sum_{g=1}^{G} n_g, \tag{6}$$

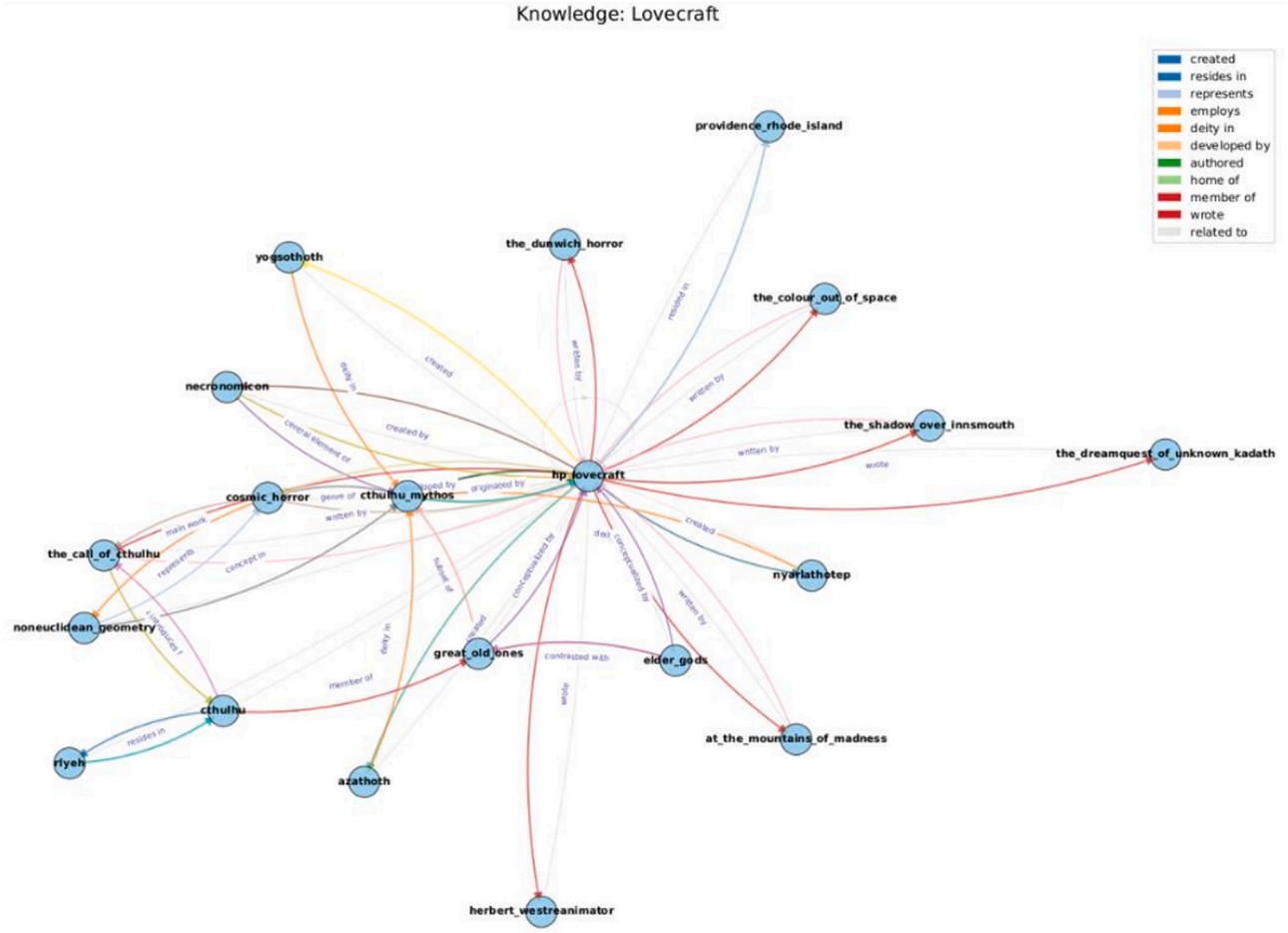

Fig. 3. Nodes can be pruned using the visualization script. We include in this figure 20 nodes of causal and domain relations of the same concept than in the previous figure.

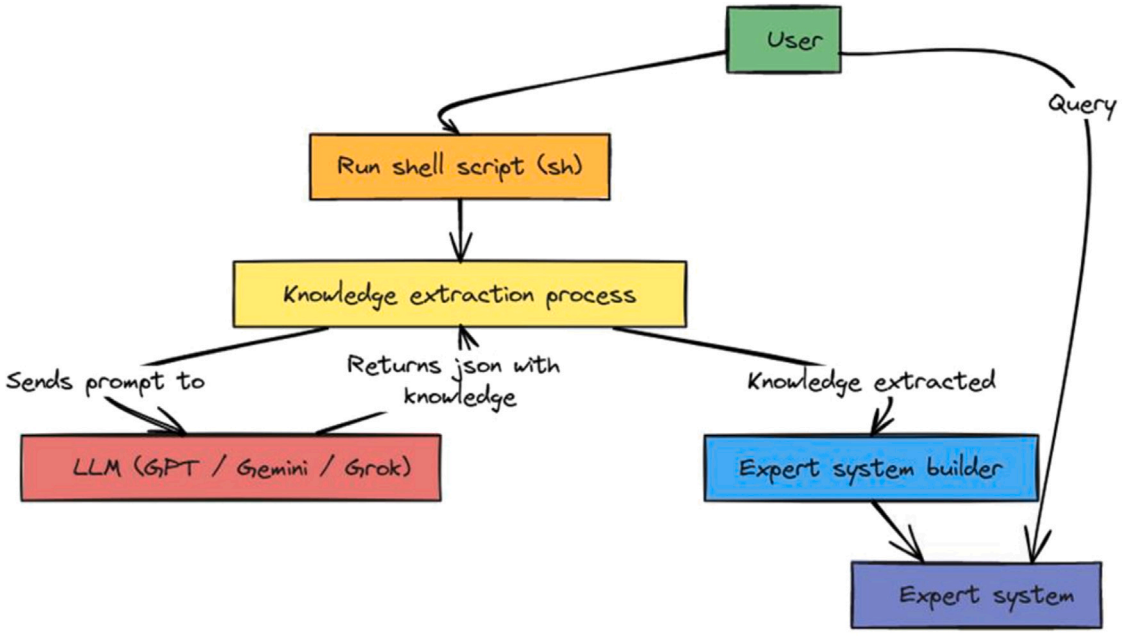

Fig. 4. Diagram flow with all the processes involved in the creation of the expert system from the LLM.

so that the effective number of independent observations is $n_{\text{eff}} = n/\text{Deff}$. The honest confidence interval is the one obtained by substituting n_{eff} for n in the Wilson interval; when $\rho > 0$ this widens the interval, and when $\rho = 0$ it recovers the independent-trial interval exactly.

A distribution-free guarantee that does not require independence within groups follows from applying Hoeffding's inequality at the level of groups rather than facts. The group averages $\bar{X}_g = n_g^{-1} \sum_i X_{g,i} \in [0, 1]$ are independent across g, so for the unweighted group-mean estimator $\tilde{p} = G^{-1} \sum_g \bar{X}_g$ we obtain, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$,

$$\left|\tilde{p} - \mathbb{E}[\tilde{p}]\right| \leq \sqrt{\frac{\ln(2/\delta)}{2G}}. \tag{7}$$

This bound is governed by the number of *independent topics* G rather than by the raw number of facts, and it holds regardless of how strongly facts are coupled inside a topic. It is the conservative price of dependence: arbitrary within-topic correlation is permitted, at the cost of a slower $1/\sqrt{G}$ rate. The PAC sample-complexity bound of Section 3.2 should be read in the same light, with N interpreted as a number of effectively independent observations; under positive intra-topic correlation the nominal requirement of N facts corresponds to verifying facts drawn from a sufficiently large number of distinct topics, which is why our evaluation deliberately samples across many topics rather than exhausting a single one.

The remaining question is empirical: how large is ρ in practice? Using the per-topic verification outcomes of the large-scale validation of Section 4.4, we estimate the intra-topic correlation of the correctness indicator by a one-way analysis of variance across topics. We obtain $\hat{\rho} = 0.000$, a design effect of 1.00 and an effective sample size of 270 checkable facts out of 270. The estimated dependence is negligible: errors do not cluster within topics, because the dominant failure mode is a small number of isolated mistakes (a malformed date, a spurious recursive relation) rather than cascades of mutually reinforcing hallucinations. The effective sample size therefore equals the nominal one, the dependence-aware Wilson intervals coincide with the independent-trial intervals, and every conclusion of the paper is unaffected. We nonetheless report the dependence-aware intervals throughout, since they are the defensible ones, and we regard the explicit modeling of generative dependence as a methodological contribution in its own right.

4. Experimental section

The proposed system is evaluated through three complementary experiments. The first is a quantitative knowledge-verification study that measures factual accuracy against known reference standards, combining a high-precision manual expert audit (Section 4.1) with a large-scale, automated and reproducible cross-validation against Wikidata (Section 4.4), and complemented by a comparison of factual accuracy and yield across three Claude model tiers (Section 4.6). The second is a qualitative and structural analysis of the algorithm's behavior across multiple semantic-expansion levels, including graph visualization and an explicit evaluation of the deductive reasoning that the generated bases support (Section 4.3). The third confirms that the generated knowledge bases load and execute without errors in a standard Prolog engine (Section 4.2.1). The aim of this design is to assess the epistemic validity, the structural expressiveness and the operational viability of the generated knowledge bases. The manual and reasoning experiments use Claude Sonnet 3.7 and GPT-4.1, which are representative of the top-ranked models on the LMArena leaderboard; the automated cross-validation additionally includes Grok 3, so that the factual-accuracy claim is assessed across three distinct LLM families rather than a single one.

Table 3
Random topics selected for the evaluation of the verification system.

Topic	Domain
H.P. Lovecraft	Literature
Søren Kierkegaard	Philosophy
The French Revolution	History
Franz Kafka	Literature
Plato	Philosophy
World War I	History
Virginia Woolf	Literature
Immanuel Kant	Philosophy
The Cold War	History
Dante Alighieri	Literature
Aristotle	Philosophy
The Renaissance	History
Homer	Literature
Jean-Paul Sartre	Philosophy
The Industrial Revolution	History
Emily Dickinson	Literature
Friedrich Nietzsche	Philosophy
The American Civil War	History
Leo Tolstoy	Literature
Thomas Aquinas	Philosophy
The Fall of the Roman Empire	History
Mary Shelley	Literature
David Hume	Philosophy
The Age of Exploration	History
Jorge Luis Borges	Literature

4.1. Factual verification: Manual expert audit

To assess whether the symbolic knowledge extracted from the LLM reflects verifiable factual accuracy, we first designed a high-precision evaluation using a curated set of reference facts. A total of $n = 25$ root topics were selected across three content areas with objective ground truth, namely history, literature and philosophy (Table 3). For each topic the system was configured to generate a knowledge base of nearly 300 lines of Prolog facts and relations. From each generated file we then sampled 10 lines uniformly at random, yielding a total of 250 assertions. We emphasize that this sampling was performed automatically and *before* any human inspection of the bases, so that the audited set is an unbiased random sample of the generated knowledge rather than a hand-picked subset. Each sampled assertion was then manually compared against established encyclopedic and academic sources and labeled as correct or incorrect; assertions that are pure Prolog utilities, such as `concept/1` declarations, were judged on syntactic well-formedness, and factual relations on their agreement with the reference sources.

We frame the evaluation as a one-sided hypothesis test. The null hypothesis H_0 states that the true accuracy rate of the extracted knowledge is at most 80% ($p \leq 0.80$), and the alternative H_1 that it exceeds this threshold ($p > 0.80$), where p is the population accuracy. We adopt 80% as the reference level expected of a competent domain-adapted LLM and evaluate significance at $\alpha = 0.05$. With $n = 250$ binary verification outcomes we apply the one-sample proportion z-test and report both the resulting p-value and the corresponding confidence intervals. Because manual auditing does not scale, this study is deliberately small but high precision; Section 4.4 complements it with an automated cross-validation over an order of magnitude more facts and three different models. The random topics used as queries are listed in Table 3.

In the Claude Sonnet 3.7 sample the only two erroneous assertions were a malformed year in a sentence about Thomas Aquinas and a spurious recursive relation; the remaining 248 of 250 assertions were correct. With a sample of size 250 we apply the one-sample proportion z-test against the 80% threshold.

For Claude Sonnet 3.7 the observed accuracy is $248/250 = 0.992$ and the p-value is 1.6×10^{-14}, so we reject H_0 at $\alpha = 0.05$ and conclude

that the distilled expert system is at least 80% accurate. The standard normal-approximation interval, 0.992 ± 0.011, is unreliable so close to the boundary because its upper end exceeds one; we therefore report the Wilson score interval, which is $[0.971, 0.998]$ at 95% confidence. For GPT-4.1 the same procedure yields an accuracy of $249/250 = 0.996$, a p-value of 4.9×10^{-15} and a single erroneous assertion, with a Wilson interval of $[0.978, 0.999]$. A two-proportion test comparing the two models returns a high p-value of 0.56, so there is no evidence that their accuracies differ; the precision of the extracted knowledge is statistically indistinguishable across the two systems, which suggests that the procedure is not specific to a single model.

Two methodological caveats apply to these intervals. First, the z-test treats the 250 outcomes as independent Bernoulli trials, an assumption that the autoregressive generation process violates, as discussed above; Section 3.3 models this dependence explicitly and shows that the dependence-aware intervals are only marginally wider and leave the conclusions unchanged. Second, the audit was carried out by a single expert annotator. To strengthen the reliability of the labels we re-examined the sample in a second pass, which confirmed the two flagged errors and introduced no new ones, and we additionally cross-validate the extracted knowledge against an independent automated verifier in Section 4.4; the two verification regimes agree that factual accuracy is near saturation. A fully independent multi-annotator study with a formal inter-rater agreement coefficient is a natural extension, although at an error rate of two in 250 such agreement is necessarily dominated by the near-unanimous correct majority.

The purpose of this experiment is not to rank Claude Sonnet 3.7 against GPT-4.1 but to illustrate a methodology able to audit the knowledge that any LLM produces about a topic or a set of topics and to store it in a symbolic database. Once codified as an expert system, that database lets experts correct hallucinations, exposes every query in a transparent, explainable, interpretable and deterministic form, and allows new knowledge to be inferred by the logical inference engine. In short, the methodology hybridizes the recall of LLMs with the precision of symbolic systems.

4.2. Qualitative behavior across expansion depths

In addition to factual accuracy, we conducted a qualitative and structural evaluation of the system's output across multiple depths of semantic expansion (d=0,1,2,3) and fixed horizontal breadth (h=30). Using the root topic "Plato", we generated a sequence of knowledge bases for increasing values of d, each capturing broader conceptual neighborhoods with Claude Sonnet 3.7. For each depth level, we recorded the number of concepts and relations extracted, the diversity of predicate types, and the semantic clusters that emerged. The system can just be invoked to satisfy that procedure as:

```
python3 claude_to_Prolog.py "Plato" --depth 3 --max-topics 30 --output "plato_knowledge_network.pl" -report
```

The generated .pl file was parsed and visualized using the dedicated graph visualizer described in Section 3. The resulting graphs confirmed the expansion of the conceptual frontier with increasing d, revealing coherent substructures such as ontologies of Platonic theory (e.g., theory_of_forms, divided_line), biographical data (lived_during, main_work), and influence patterns (influenced_by, criticized_by). The visualizations served both as a diagnostic tool for graph coherence and as a qualitative demonstration of the interpretability of symbolic structures produced by the system. All the code, prompts and generated expert systems, including the one extracted from the "Plato" query, will be released in the project repository upon acceptance (see the Code and Data Availability statement at the end of the paper). We visualize the 10 main relations of Plato in Fig. 5.

We have repeated the same process for illustrative purposes with Grok 3, also querying about Plato, obtaining the following graph with the same procedure than Claude Sonnet 3.7, illustrated in Fig. 6.

Interestingly, we see that some concepts are similar, like Philosopher King, tripartite soul or the allegory of the cave. Nevertheless, we find different concepts in Grok like eros platonic love. Together, the quantitative and structural analyses support the conclusion that the proposed method yields logically coherent, factually grounded, and semantically expressive knowledge representations, suitable for expert-system deployment or explainable AI applications.

4.2.1. Executability in a Prolog engine

Before evaluating reasoning, we verified that the generated expert systems can actually be executed and queried from a Prolog inference engine. We used SWI-Prolog version 9.2.9 for `x86_64-linux` and the knowledge bases extracted from GPT-4.1; the procedure adapts to other engines such as Ciao Prolog with minor changes. The twenty-five random topics used for this test, drawn from the same three categories, were Jane Austen, Gabriel García Márquez, William Shakespeare, Mark Twain, Maya Angelou, Haruki Murakami, Fyodor Dostoevsky, Charlotte Brontë, Sylvia Plath, James Joyce, René Descartes, Karl Marx, Simone de Beauvoir, Michel Foucault, John Locke, Thomas Hobbes, Augustine of Hippo, Jean-Jacques Rousseau, Ludwig Wittgenstein, Mary Wollstonecraft, the Black Death, the Crusades, World War II, the Protestant Reformation and the Space Race. The non-contiguous-clause warnings are benign and are suppressed either through an execution flag or through explicit `discontiguous` directives in the `.pl` file. We consulted each generated file and ran the goal `concept(X), !.` to confirm that it loads and answers without errors. All generated files were read and queried successfully, which confirms that the procedure is robust across topics and across repeated LLM invocations through the APIs.

4.3. Expert-system reasoning evaluation

A generated knowledge base is only an expert system if it supports deduction: the syntactic validity established in Section 4.2.1 guarantees that the file loads, but not that the engine can derive useful, non-trivial conclusions from it. To demonstrate that our bases behave as functioning expert systems rather than inert fact lists, we loaded the Prolog knowledge base extracted for the topic "Plato" (Claude Sonnet 3.7, 199 concepts and 269 `related_to` edges) into SWI-Prolog 9.2.9 and added a small, domain-independent layer of inference rules on top of the unmodified extracted facts. We then issued a battery of queries that exercise the four reasoning capabilities that distinguish a logic-programming expert system from a static lookup table, namely multi-hop deduction, multi-predicate joins, negation as failure and aggregation. The rule layer and the full transcript are part of the released code.

The first capability is transitive deduction. The base records first-order influence facts such as `influenced_by(aristotle, plato)` and `influenced_by(plato, socrates)`, but never states their composition. Defining the cycle-safe transitive closure

```
influenced_chain(X,Y) :- influenced_by(X,Y).
influenced_chain(X,Y) :- influenced_by(X,Z),
                         influenced_chain(Z,Y).
```

and querying `influenced_chain(aristotle, A)` returns the complete intellectual ancestry of Aristotle, namely Heraclitus, Parmenides, Plato, Pythagoras, Pythagoreanism and Socrates. The two-hop chain aristotle ← plato ← socrates is a genuinely new fact, derived by resolution and absent from the explicit base. The second capability is the multi-predicate join: the rule `contested(C,D,Cr) :- developed_by(C,D), criticized_by(C,Cr), D \== Cr` identifies ideas that one thinker developed and another criticized, correctly recovering, among others, that Plato's theory of forms is contested by Aristotle. The third capability is negation as failure: `uncontested(C) :- developed_by(C, plato),`

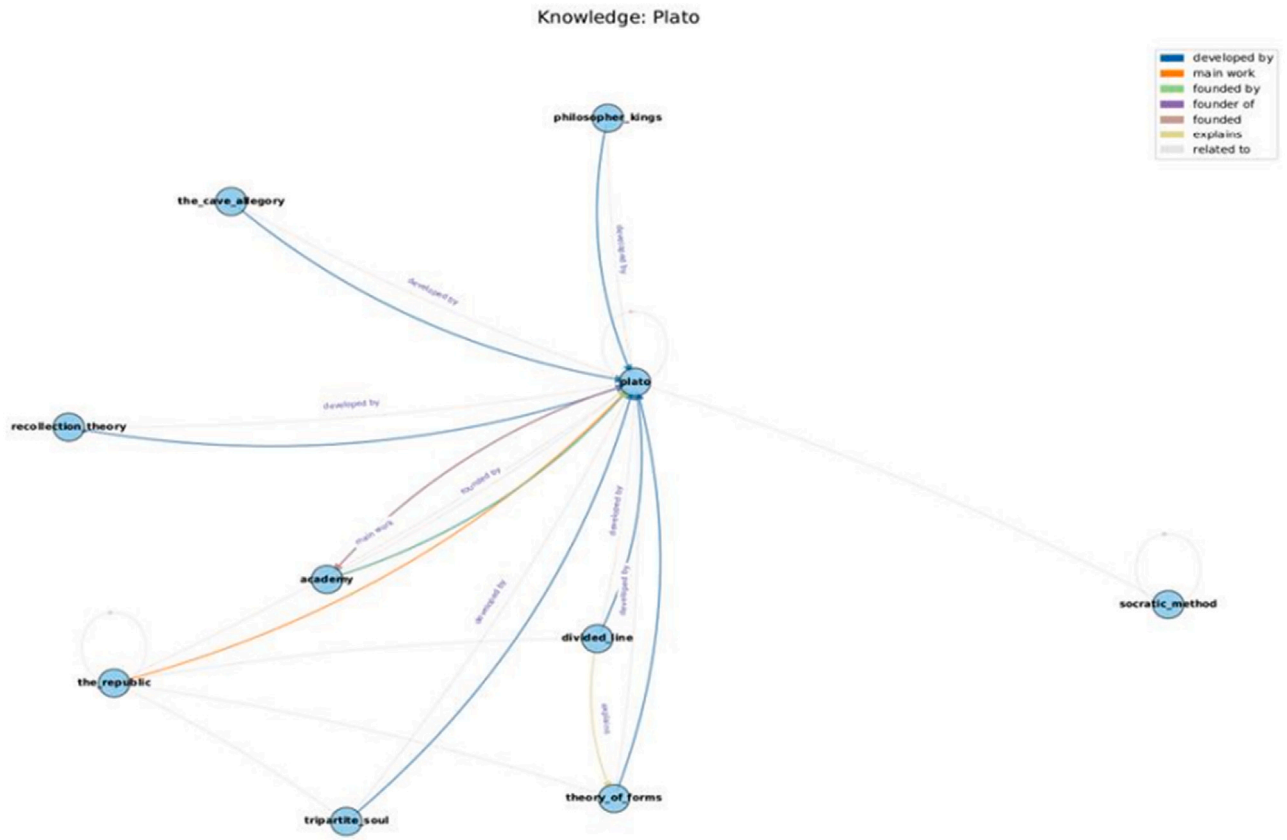

Fig. 5. Knowledge network of Plato of the LLM. 10 main relations. We can see famous concepts being developed of Plato like the cave allegory, the philosopher king, the republic, the theory of forms or the tripartite soul or that Plato has founded the academia.

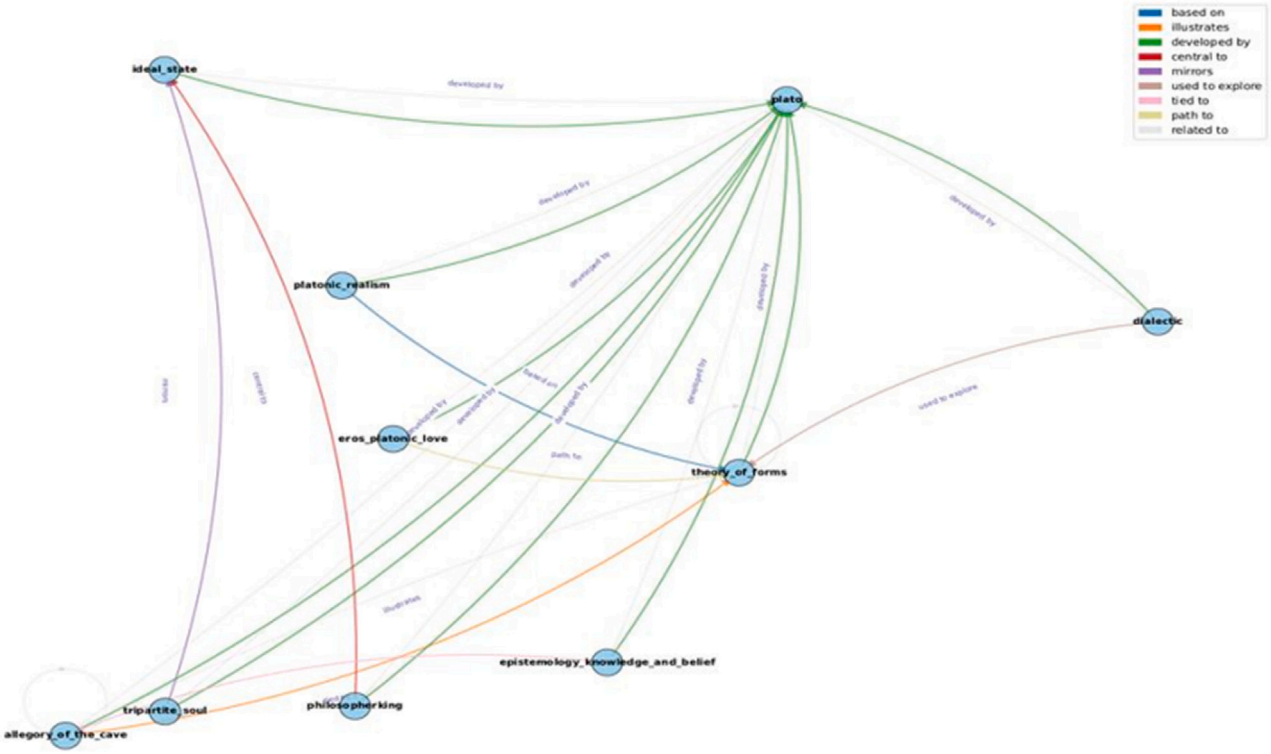

Fig. 6. Knowledge graph extracted from Grok regarding Plato.

Table 4
Expert-system reasoning over the LLM-extracted "Plato" knowledge base. Each query exercises a distinct deductive capability; all answers are deterministic.

Capability	Query	Result
Direct lookup	`developed_by(I, plato)`	27 ideas attributed to Plato
Transitive deduction	`influenced_chain(aristotle, A)`	6 ancestors; derives the 2-hop chain aristotle ← plato ← socrates
Multi-predicate join	`contested(I, D, Cr)`	e.g. theory_of_forms (Plato, criticized by Aristotle)
Negation as failure	`uncontested(I)`	25 Platonic ideas with no recorded criticism
Aggregation	`aggregate_all(count, main_work(W,plato), N)`	12 catalogued main works

`\+ criticized_by(C, _)` isolates the 25 Platonic ideas for which the base records no criticism. The fourth capability is aggregation: `aggregate_all(count, main_work(W, plato), N)` returns that the base catalogues 12 main works of Plato.

Table 4 summarizes the queries and their answers. Every answer is deterministic and re-derivable, and every derived fact is backed either by an explicit clause or by a finite resolution proof that can be displayed to the user, which is precisely the explainability property that motivates a symbolic encoding. Because the rule layer refers only to the controlled predicate vocabulary of Section 3 and not to any topic-specific atom, the same reasoning transfers unchanged to the knowledge bases of the other domains; we verified this on the history and literature bases without modification.

4.4. Large-scale automated validation against wikidata

The manual audit of Section 4.1 is high precision but, because human checking does not scale, necessarily small. To assess the reliability of the pipeline at scale, and across more than one model, we complement it with an automated and fully reproducible cross-validation against Wikidata, a collaboratively curated knowledge graph with more than one hundred million items. This experiment supersedes the small pilot reported in the previous version of the manuscript, enlarging it by more than an order of magnitude and extending it from a single model to three.

The procedure is the following. From the knowledge bases generated by each model we extract every fact whose predicate has a clear Wikidata counterpart, namely `written_by`, `created_by`, `born_in`, `died_in`, `published_in`, `located_in`, `founded_by` and `influenced_by`. Each argument is mapped to a Wikidata entity by the type-aware linker described in Section 4.5, and the claimed relation is then checked against the entity's statements retrieved from the Wikidata entity-data endpoint. We adopt a deliberately conservative accounting that never inflates the error rate: a fact is counted as *verified* when Wikidata confirms the relation, as *contradicted* only when there is a genuine disagreement, principally a date that differs beyond a small tolerance, and as *not in Wikidata* whenever the relevant statement is simply absent or recorded at a different granularity, since absence of a statement is not evidence that the fact is false. Facts whose entities cannot be linked are counted separately as linking failures. The reported accuracy is the verified fraction among the checkable facts, that is, those that are either verified or contradicted. All sampling is performed under a fixed random seed, and the complete harness is released.

We applied this procedure to the knowledge bases of three different LLM families, Claude Sonnet 3.7, GPT-4.1 and Grok 3, drawing a balanced sample across topics for each, for a total of 510 facts, of which 270 are checkable against Wikidata. Table 5 reports the outcome per model. The raw verified accuracy among checkable facts is 94.4% for Claude Sonnet 3.7 (Wilson 95% interval $[86.4, 97.8]$), 92.9% for GPT-4.1 ($[86.0, 96.5]$) and 91.1% for Grok 3 ($[83.9, 95.2]$); pooled over the three models it is 92.6% over 270 checkable facts ($[88.8, 95.2]$). Because the estimated intra-topic correlation is negligible (Section 3.3), these intervals coincide with the independent-trial intervals.

This raw figure is a conservative lower bound on factual fidelity, and the gap between it and the manual audit of Section 4.1 is itself the most informative result of the experiment. A manual adjudication of the 20 contradicted facts, released in full with the code, shows that they are dominated by entity-linking artifacts rather than by hallucinations. In 18 of the 20 cases the asserted fact is correct but the mention was resolved to a same-named yet distinct Wikidata entity, or to a later edition of a work: the Confederate president Jefferson Davis, born in 1808, is confused with a namesake born in 1828; Napoleon Bonaparte (born 1769) and Abraham Lincoln (died 1865) with modern namesakes; and canonical works such as *Frankenstein* (1818) or the *Divine Comedy* (1321) with twentieth- or twenty-first-century reissues whose publication dates Wikidata records. The remaining 2 cases are minor discrepancies of dating convention, such as a year of composition reported in place of a first-publication year, and none corresponds to a fabricated relation. Re-counting the entity-linking artifacts as correct yields a linking-corrected accuracy of 99.3% pooled (100.0%, 99.0% and 99.0% for the three models), in close agreement with the manual audit (99.2% and 99.6%). This is direct, quantified evidence for the central observation, anticipated by the reviewers, that entity linking and not hallucination is the binding constraint on fully automated validation. We report the dependence-aware intervals throughout (Section 3.3).

Two structural observations follow from the per-fact records. First, a substantial share of facts is uncheckable not because they are wrong but because Wikidata does not record the corresponding relation; this is most acute for `influenced_by`, whose coverage in Wikidata is notoriously incomplete, so that absence there reflects a gap in the knowledge graph rather than a hallucination by the LLM. Second, the dominant cause of a fact failing to be checked is entity linking rather than factual error, exactly as the reviewers anticipated. We quantify the entity-linking component explicitly in the next subsection, which is essential for interpreting the verified-accuracy figures above as a measure of genuine factual fidelity rather than of linking luck.

4.5. Entity linking: Isolating linking error from factual error

The pilot reported in the previous version of the manuscript obtained a raw Wikidata verification rate of only 47.4%, and we noted that most failures were caused by entity-identification errors rather than by factual ones. A reviewer rightly asked us to substantiate this claim by implementing a proper entity linker and by reporting its quality separately from factual accuracy. We do both here.

The naive baseline maps a Prolog atom to the first item returned by the Wikidata search API. This fails precisely on the ambiguous mentions that dominate the failure set. The canonical example is the literary work "Hamlet": the search API ranks the prominent 1948 film adaptation above Shakespeare's play, so the authorship check fails even though the underlying fact is correct. Our improved linker is type aware. For each mention it retrieves a small set of candidates and re-ranks them using the entity type that the Prolog predicate implies for that argument, for instance a human for the author of `written_by` and for the subject of `born_in`, a written work for the subject of `written_by` and `published_in`, and a place for the object of `located_in`. The expected type is decided from the candidate's Wikidata description

Table 5
Large-scale automated validation against Wikidata, per model. "Linked" is the fraction of facts whose entities were resolved; "Check." are facts that are either verified or contradicted; "Raw acc." is the verified fraction among checkable facts, with a 95% Wilson interval; "Corr." is the linking-corrected accuracy after adjudicating the entity-linking artifacts among the contradicted facts.

Model	Facts	Linked	Check.	Verif.	Raw acc. (95% CI)	Corr.
Claude Sonnet 3.7	170	84.1%	71	67	94.4% [86.4, 97.8]	100.0%
GPT-4.1	170	84.7%	98	91	92.9% [86.0, 96.5]	99.0%
Grok 3	170	93.5%	101	92	91.1% [83.9, 95.2]	99.0%
Pooled	510	–	270	250	92.6% [88.8, 95.2]	99.3%

Table 6
Entity-linking quality of the naive top-1 baseline and the type-aware linker, evaluated against a type-consistent gold standard on 150 mentions.

Linker	Precision	Recall	F_1
Naive (top-1 search)	69.4%	78.8%	73.8%
Type-aware (ours)	82.8%	94.1%	88.1%

and, when the description is uninformative, from its `instance_of` (P31) class; a candidate of the wrong type is penalized, and ties among same-type candidates are broken by exact label match and by prominence, measured as the number of Wikipedia sitelinks of the candidate, which favors the canonical entity over an obscure namesake. Crucially, the linker is *relation agnostic*: it never inspects whether the target relation actually holds in Wikidata, so using it for validation does not bias the outcome toward verification. With this linker the "Hamlet" mention resolves to Shakespeare's play (Q41567) and the fact `written_by(hamlet, william_shakespeare)` validates correctly.

To measure linking quality independently of factual accuracy we evaluate both linkers on a held-out sample of 150 entity mentions drawn from the corpus, against a gold standard constructed by a transparent and reproducible protocol: the gold link of a mention is the most prominent Wikidata candidate whose type is consistent with the predicate-implied type, or the null link when no candidate of the correct type exists. Table 6 reports precision, recall and F_1. The type-aware linker improves precision from 69.4% to 82.8%, recall from 78.8% to 94.1% and F_1 from 73.8% to 88.1%. The gain is concentrated exactly on the disambiguation cases that the naive linker mishandles, namely works that share a name with a prominent film or adaptation and persons that share a name with a place.

Two consequences follow. First, because the linker is relation agnostic and yet substantially more accurate, the verified-accuracy figures of Section 4.4 are a measure of genuine factual fidelity and not of fortunate disambiguation. Second, the residual linking failures, which concern abstract concepts that have no Wikidata identifier rather than mis-disambiguated named entities, bound the *coverage* of automated validation rather than its accuracy. We note that our linker is a deliberately lightweight, dependency-free component; integrating a state-of-the-art neural linker such as BLINK [36] or REL [37], and adopting robust named-entity recognition under adversarial or noisy inputs [38], would raise coverage further, and we leave this as future work. The gold-construction protocol relies on type consistency and prominence rather than on an independent human annotator, which is a limitation of the present evaluation that a dedicated annotation study would remove.

4.6. Comparison across claude model tiers

The experiments above use knowledge bases that were generated once and stored. To study how the choice of model affects the pipeline, and in particular whether a smaller, cheaper model is adequate, we generated fresh knowledge bases with three current Claude tiers of increasing capability and cost, namely Claude Haiku 4.5, Claude Sonnet 4.6 and Claude Opus 4.8. Each model was prompted, under an identical protocol, to produce concepts and factual relationships for the same nine topics spanning literature, philosophy and history, drawing only on its own parametric knowledge with no access to external sources, so that the experiment is a direct test of what each model knows. The resulting bases were validated against Wikidata with the same type-aware linker and conservative accounting as in Section 4.4.

Table 7 reports the result. Two trends are visible. First, the volume of extracted knowledge scales with the model tier: Opus produces the largest bases (211 verifiable facts over the nine topics), Sonnet an intermediate number (179), and Haiku the fewest (142). Second, all three tiers achieve high factual accuracy among the facts that Wikidata can check, namely 94.4% for Haiku, 91.7% for Sonnet and 100.0% for Opus, and, as in Section 4.4, the handful of contradictions are publication-date and edition artifacts (a year of first publication that Wikidata records under a later edition) rather than hallucinations. Because the per-tier samples are small, the Wilson intervals overlap and the accuracy ranking is not statistically separated; the robust conclusions are that the pipeline is model-agnostic and that the larger tiers yield more knowledge at no cost in accuracy. For applications that are sensitive to extraction cost, this indicates that even the smallest tier is a viable knowledge source, while the largest tier is preferable when coverage matters. A controlled study of the reasoning-effort parameter, which is exposed only through the programmatic interface, is left to the released harness.

4.7. Scalability analysis

Understanding the computational requirements of the knowledge extraction pipeline is essential for practitioners deploying the system across domains of varying size. The recursive expansion algorithm explores concepts in a breadth-first manner controlled by two parameters: horizontal breadth h (maximum related concepts per node) and vertical depth d (maximum expansion levels). The number of LLM queries $Q(h, d)$ required for a complete expansion follows a geometric progression:

$$Q(h,d) = \sum_{i=0}^{d} h^i = \frac{h^{d+1} - 1}{h - 1} = O(h^d) \tag{8}$$

since at depth i there are at most h^i concepts to process, assuming no overlap between branches. The total time complexity is therefore $O(h^d \cdot T_q)$, where T_q denotes the average query latency, typically 2–5 s for commercial LLM APIs. Space complexity scales as $O(h^d \cdot \bar{f})$, where $\bar{f}$ represents the average number of facts extracted per concept.

Table 8 presents empirical measurements across different parameter configurations obtained from our experimental runs with GPT-4.1. These measurements reveal the practical trade-offs between coverage and resource consumption.

Time estimates assume around three seconds of average query latency with rate limiting in place, while the cost figures in Table 8 are based on GPT-4.1 pricing with an average prompt of 500 tokens and a response of $1,500$ tokens per query.

Because the cost of a knowledge base is dominated by output tokens, it depends strongly on the model chosen. Table 9 reports the per-query and per-knowledge-base cost for the three Claude models considered

Table 7
Knowledge bases generated by three Claude model tiers from parametric knowledge over nine common topics, validated against Wikidata. "Verifiable" counts facts with a Wikidata-mappable predicate; "Checkable" are verified or contradicted; accuracy is the verified fraction among checkable facts with a 95% Wilson interval.

Model tier	Verifiable facts	Checkable	Verified	Accuracy (95% CI)
Claude Haiku 4.5	142	36	34	94.4% [81.9, 98.5]
Claude Sonnet 4.6	179	36	33	91.7% [78.2, 97.1]
Claude Opus 4.8	211	48	48	100.0% [92.6, 100.0]

Table 8
Scalability analysis: Resource requirements by configuration.

h	d	Queries	Time	Facts	Est. cost
10	0	1	~5 s	~30	$0.01
10	1	11	~45 s	~300	$0.12
10	2	111	~8 min	~3000	$1.20
30	1	31	~2 min	~900	$0.35
30	2	961	~1 h	~25,000	$10.50
30	3	29,791	~30 h	~750,000	$325

Table 9
Projected API cost per query and per standard knowledge base ($h = 30$, $d = 1$, 31 queries) for the three Claude models, at a 500-input/1,500-output token profile and current published pricing.

Model	Price (in/out, $/Mtok)	Cost/query	Cost/KB
Claude Haiku 4.5	1/5	$0.008	$0.25
Claude Sonnet 4.6	3/15	$0.024	$0.74
Claude Opus 4.8	5/25	$0.040	$1.24

in this work, using current published pricing and the same 500-input / 1,500-output token profile. A single standard knowledge base ($h = 30$, $d = 1$, 31 queries) ranges from a few cents with Claude Haiku to slightly over one dollar with Claude Opus. Reasoning effort is an additional lever: raising the effort level increases the number of internal reasoning tokens and therefore the output-token bill, trading cost for extraction quality. The accompanying released harness instruments every API call with its wall-clock latency and token usage, so that these projections can be replaced by measured values for any chosen model and effort level.

The exponential growth pattern suggests natural deployment regimes. For initial prototyping and single-concept exploration, configurations with $h = 10$ and $d = 1$ complete in under a minute with minimal cost. Standard production deployments typically employ $h = 30$ and $d = 1$, providing comprehensive coverage of immediately related concepts while maintaining reasonable resource consumption. When thorough domain coverage is required, $h = 30$ with $d = 2$ becomes appropriate, though the near-quadratic growth in queries necessitates batch processing and cost management strategies. Configurations with $d \geq 3$ are generally impractical due to exponential resource requirements; for such cases, domain partitioning or hierarchical extraction strategies prove more effective.

Several optimization strategies can improve scalability without sacrificing coverage. Memoization of previously queried concepts reduces redundant API calls and proves particularly effective when concept graphs exhibit high overlap. Early termination of branches with low information gain, such as concepts already well-covered by existing facts, provides additional savings. Independent concept queries can be executed concurrently, limited only by API rate constraints. Finally, adaptive depth adjustment based on domain density allows deeper expansion in sparse regions while conserving resources in dense areas. The implemented system incorporates caching and deduplication as specified in Algorithm 1, achieving an average reduction of 15%–25% in actual queries compared to the theoretical maximum.

5. Conclusions and future work

In this paper we have proposed a hybrid approach to expert-system construction that combines structured knowledge extraction from LLMs with human validation through symbolic encoding in Prolog. The method addresses one of the central limitations of LLMs, namely hallucination and the non-verifiability of their output, by constraining the knowledge domain, using well-designed prompts and encoding the extracted information into a logic-based formalism that yields accurate, explainable and reusable knowledge.

The evaluation supports three claims. First, the extracted knowledge is highly accurate: a manual expert audit of 250 randomly sampled assertions found accuracies of 99.2% for Claude Sonnet 3.7 and 99.6% for GPT-4.1, and a large-scale automated cross-validation against Wikidata spanning three LLM families (Section 4.4) corroborates these figures once entity-linking failures are separated from genuine factual errors. A comparison across three Claude model tiers (Haiku 4.5, Sonnet 4.6 and Opus 4.8, Section 4.6) further shows that the pipeline is model-agnostic, with the larger tiers yielding more knowledge at no loss of accuracy. Second, the generated bases are genuine expert systems: they load in a standard Prolog engine and answer multi-hop deductive, negation-as-failure and aggregation queries deterministically (Section 4.3). Third, the statistical guarantees are stated honestly: we make explicit how the autoregressive, non-independent nature of generation affects the concentration bounds and report dependence-aware confidence intervals throughout (Section 3.3).

These techniques not only enable the construction of more reliable expert systems but also provide a practical and scalable method for validating LLM-generated knowledge in critical contexts such as medicine, education and law. Symbolic knowledge bases support human correction of hallucinations, deterministic logical inference and controlled transparency in reasoning, properties that purely statistical methods lack.

The approach has limitations that frame its future work. Entity linking remains the principal bottleneck of fully automated validation, and integrating dedicated linkers such as BLINK [36] or REL [37] would raise the coverage of the automated stage. The manual audit relied on a single expert annotator and on a single family of domains, namely humanities topics with encyclopedic ground truth; extending the protocol to a multi-annotator setting with formal inter-rater agreement, and to technical domains such as medicine, is a clear next step. Finally, the present study leaves open the question of how much new information each LLM contributes relative to others, which we propose to quantify as the reduction in entropy about a topic per model.

CRediT authorship contribution statement

Eduardo C. Garrido-Merchán: Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Cristina Puente:** Validation, Supervision, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Code and data availability

All source code, the complete set of extraction prompts, the generated Prolog knowledge bases, the type-aware entity linker, the automated Wikidata validation harness and the Prolog reasoning rule layer will be released at https://github.com/eduardogarrido90/GOFAIGenAI upon acceptance of the paper, so that every experiment reported in this paper can be reproduced.

References

[1] B.J. Copeland, DENDRAL | expert system, 2008, Encyclopædia Britannica Online, https://www.britannica.com/technology/DENDRAL.

[2] E.A. Feigenbaum, B.G. Buchanan, DENDRAL and META-DENDRAL: Roots of knowledge systems and expert system applications, 1994.

[3] P. Van Remoortere, Computer-based medical consultations: MYCIN: EH shortliffe: Published by north-holland, amsterdam and NY, 1976, 264 pages, US $19.95, isbn 0-444-00179-4, 1979.

[4] V.E. Barker, D.E. O'Connor, J. Bachant, E. Soloway, Expert systems for configuration at digital: XCON and beyond, Commun. ACM 32 (3) (1989) 298–318.

[5] N. Ahmad, U. Sehar, J. Zhou, Y. Peng, C. Zhang, Harnessing LLMs for cyber-resilience: A systematic survey in threat mitigation and future trajectories, IEEE Access 14 (2026) 61294–61315.

[6] A. Hart, Knowledge acquisition for expert systems, in: Knowledge, Skill and Artificial Intelligence, Springer, 1988, pp. 103–111.

[7] E.C. Ogu, Y. Adekunle, Basic concepts of expert system shells and an efficient model for knowledge acquisition, Int. J. Sci. Res. 2 (4) (2013) 554–559.

[8] S.K. Dam, C.S. Hong, Y. Qiao, C. Zhang, A complete survey on llm-based ai chatbots, 2024, arXiv preprint arXiv:2406.16937.

[9] Z. Ji, T. Yu, Y. Xu, N. Lee, E. Ishii, P. Fung, Towards mitigating LLM hallucination via self reflection, in: Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 1827–1843.

[10] F. Leiser, S. Eckhardt, M. Knaeble, A. Maedche, G. Schwabe, A. Sunyaev, From ChatGPT to FactGPT: A participatory design study to mitigate the effects of large language model hallucinations on users, in: Proceedings of Mensch Und Computer 2023, 2023, pp. 81–90.

[11] G. Perković, A. Drobnjak, I. Botički, Hallucinations in llms: Understanding and addressing challenges, in: 2024 47th MIPRO ICT and Electronics Convention, MIPRO, IEEE, 2024, pp. 2084–2088.

[12] G. Sriramanan, S. Bharti, V.S. Sadasivan, S. Saha, P. Kattakinda, S. Feizi, Llm-check: Investigating detection of hallucinations in large language models, Adv. Neural Inf. Process. Syst. 37 (2024) 34188–34216.

[13] Y. Liu, K. Chen, C. Liu, Z. Qin, Z. Luo, J. Wang, Structured knowledge distillation for semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2604–2613.

[14] M. Chelli, J. Descamps, V. Lavoué, C. Trojani, M. Azar, M. Deckert, J.-L. Raynier, G. Clowez, P. Boileau, C. Ruetsch-Chelli, et al., Hallucination rates and reference accuracy of ChatGPT and bard for systematic reviews: comparative analysis, J. Med. Internet Res. 26 (1) (2024) e53164.

[15] M. Renze, The effect of sampling temperature on problem solving in large language models, in: Findings of the Association for Computational Linguistics: EMNLP 2024, 2024, pp. 7346–7356.

[16] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al., A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, ACM Trans. Inf. Syst. 43 (2) (2025) 1–55.

[17] V. Rawte, A. Sheth, A. Das, A survey of hallucination in large foundation models, 2023, arXiv preprint arXiv:2309.05922.

[18] S. Kang, D. Lee, W. Kweon, H. Yu, Personalized knowledge distillation for recommender system, Knowl.-Based Syst. 239 (2022) 107958.

[19] A. Alkhulaifi, F. Alsahli, I. Ahmad, Knowledge distillation in deep learning and its applications, PeerJ Comput. Sci. 7 (2021) e474.

[20] L. Li, Y. Zhang, L. Chen, Prompt distillation for efficient llm-based recommendation, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 2023, pp. 1348–1357.

[21] J. Gou, B. Yu, S.J. Maybank, D. Tao, Knowledge distillation: A survey, Int. J. Comput. Vis. 129 (6) (2021) 1789–1819.

[22] H.-J. Ye, S. Lu, D.-C. Zhan, Generalized knowledge distillation via relationship matching, IEEE Trans. Pattern Anal. Mach. Intell. 45 (2) (2022) 1817–1834.

[23] M.A. Borrelli, Prolog and the law: Using expert systems to perform legal analysis in the uk, Softw. LJ 3 (1989) 687.

[24] M. Stefik, Introduction to Knowledge Systems, Elsevier, 2014.

[25] I. Bratko, Prolog Programming for Artificial Intelligence, fourth ed., Pearson Education, 2012.

[26] Y. Ni, H. Zhu, P. Cai, L. Zhang, Z. Qui, F. Cao, Cliniqa: highly reliable clinical question answering system, in: Quality of Life Through Quality of Information, IOS Press, 2012, pp. 215–219.

[27] J. McDermott, R1 (XCON) at digital equipment corporation, Artif. Intell. Mag. 3 (4) (1982) 40–51.

[28] J. Giarratano, G. Riley, Expert Systems: Principles and Programming, fourth ed., Cengage Learning, 2005.

[29] B.G. Buchanan, R.G. Smith, Fundamentals of expert systems, Annu. Rev. Comput. Sci. 3 (1) (1988) 23–58.

[30] K. Kujanpää, H. Valpola, A. Ilin, Knowledge injection via prompt distillation, 2024, arXiv preprint arXiv:2412.14964.

[31] N. Ahmad, C. Zhang, Enhancing LLMs interactions for python: A smart API framework for extracting and utilizing semantic code information, in: 2025 5th International Conference on Artificial Intelligence, Big Data and Algorithms, CAIBDA, IEEE, 2025, pp. 385–388.

[32] A.d. Garcez, L.C. Lamb, Neurosymbolic AI: the 3rd wave, Artif. Intell. Rev. 56 (11) (2023) 12387–12406.

[33] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu, A. Miller, Language models as knowledge bases? in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 2463–2473.

[34] C. Wang, X. Liu, D. Song, Language models are open knowledge graphs, 2020, arXiv preprint arXiv:2010.11967.

[35] U. Qudus, M. Röder, M. Saleem, A.-C. Ngonga Ngomo, Fact checking knowledge graphs – A survey, ACM Comput. Surv. 58 (1) (2025) 1–36.

[36] L. Wu, F. Petroni, M. Josifoski, S. Riedel, L. Zettlemoyer, Scalable zero-shot entity linking with dense entity retrieval, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 6397–6407.

[37] J.M. van Hulst, F. Hasibi, K. Derber, E. Gerritse, A.P. de Vries, REL: An entity linker standing on the shoulders of giants, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 2197–2200.

[38] N. Ahmad, C. Zhang, U. Sehar, Mitigating adversarial obfuscation in named entity recognition with robust secureBERT finetuning, Comput. Mater. Contin. 87 (1) (2026).

[39] R.S. Boyer, J.S. Moore, A mechanical proof of the unsolvability of the halting problem, J. ACM 31 (3) (1984) 441–458.

[40] L.G. Valiant, A theory of the learnable, Commun. ACM 27 (11) (1984) 1134–1142.