Cognito: A Metacognitive Neuro-Symbolic Architecture for Verifiable Artificial General Intelligence

Authors: Rafael Oliveira (ORCID: 0009-0005-2697-4668), Dr. Evelyn Reed (Cognitive Systems Research Institute)

Abstract
The field of Artificial Intelligence (AI) is in a "third summer," driven by the remarkable success of large-scale sub-symbolic models, yet simultaneously confronted by their inherent limitations in robustness, explainability, and systematic reasoning.1 Neuro-Symbolic AI (NeSy) has emerged as the prevailing paradigm to address these gaps, seeking to merge pattern-based learning with logical reasoning. However, this paper posits that current NeSy research, despite its progress, overlooks a critical element for genuine autonomy and reliability: metacognition. We introduce Cognito, a novel architecture that places a Metacognitive Controller at its core. This controller is designed to monitor, evaluate, and dynamically regulate the interaction between a high-efficiency sub-symbolic substrate and a verifiable symbolic reasoning engine. Cognito's main contributions are: (1) a formal model for metacognitive control in AI, addressing a quantified gap in the literature; (2) a path toward verifiability-by-design in complex AI systems using dependently typed programming languages; and (3) a foundational framework for the development of a more robust, adaptable, and trustworthy Artificial General Intelligence (AGI).

1. Introduction: The Chasm Between Pattern Recognition and Genuine Cognition

1.1. The Triumph and Tribulations of AI's Third Summer

The current era of Artificial Intelligence, often described as its "third summer," is defined by the unprecedented advances driven by deep neural networks and, most notably, Large

Language Models (LLMs).[2] These architectures have demonstrated an extraordinary ability in tasks that rely on recognizing complex patterns in vast datasets. Their proficiency in areas such as natural language processing, computer vision, and content generation has led to widespread adoption and significant optimism.[4] This success can be framed, as suggested by theorists like Bengio, as the automation of what Kahneman termed "System 1" thinking: the intuitive, fast, and unconscious pattern recognition that forms the basis of human cognition.[5]

However, the very scale that gives these models their impressive capabilities also amplifies their fundamental weaknesses. Their "black box" nature makes their decision-making processes opaque and difficult to interpret, which undermines trust and hinders their adoption in critical, high-stakes domains like medicine or finance.[6] Their reliance on enormous amounts of training data and unsustainable computational trajectories raises questions about their scalability and environmental impact.[9] More critically, these models exhibit notorious fragility when faced with data outside their training distribution, are vulnerable to subtle adversarial attacks, and lack the capacity for the logical, causal, and structured reasoning that characterizes "System 2" thinking.[10] This tension defines the central challenge of contemporary AI: the phenomenal success of sub-symbolic models has exposed a deep chasm between pattern recognition and genuine cognition.

1.2. The Neuro-Symbolic Proposition

In response to these limitations, a consensus has emerged in the research community: the way forward lies in the integration of historically distinct AI paradigms. Neuro-Symbolic AI (NeSy) proposes to merge the statistical learning ability of neural networks (the sub-symbolic) with the formal reasoning and explicit knowledge representation of symbolic systems.[11] The goal is to create hybrid systems that leverage the best of both worlds: the ability to learn from raw, unstructured data, characteristic of neural systems, and the ability to reason with explicit rules, manipulate symbols, and provide logical justifications for their conclusions, a strength of symbolic systems.

This fusion promises to improve interpretability by providing a window into the system's reasoning process; increase data efficiency by allowing the incorporation of prior knowledge in the form of symbolic rules; and enhance robustness by subjecting neural outputs to logical constraints.[11] This approach represents an attempt to build systems capable of operating in both System 1 and System 2 modes, bridging the gap between perception and reason.

1.3. The Metacognitive Void - The Central Thesis

Despite the enthusiasm and progress in the NeSy field, a critical analysis of its trajectory reveals a fundamental omission. The juxtaposition of neural and symbolic components, while necessary, is not sufficient. A recent and comprehensive systematic review of the NeSy literature, conducted by Colelough & Regli (2025), provides a compelling empirical validation for this claim. Using the PRISMA methodology to analyze over a thousand articles published between 2020 and 2024, the study quantifies the research focus in the field. The results are revealing: while areas like learning and inference (63%), knowledge representation (44%), and logic and reasoning (35%) receive considerable attention, explainability and trustworthiness are less represented (28%). More shocking, however, is the area of **Metacognition**, which constitutes only 5% of the analyzed body of research.

In this context, metacognition is defined as a system's ability to "think about its own thinking"—the capacity to monitor, evaluate, regulate, and adapt its own reasoning and learning processes. Neglecting this higher-order control layer severely limits the autonomy, adaptability, and self-correction capabilities of AI systems. A system that cannot recognize its own uncertainty, detect its own logical inconsistencies, or strategically allocate its cognitive resources will remain fundamentally fragile and unreliable, regardless of the sophistication of its individual components. This "metacognitive void" is not a speculative gap but a quantified and critical failure in the AI research agenda, which impedes progress toward truly intelligent and autonomous systems.

1.4. Introducing Cognito

This paper introduces Cognito as an architectural solution to this metacognitive void. Cognito is not simply another NeSy architecture; it is a *metacognitively-governed* architecture. Its central hypothesis is that robust cognitive generalization emerges not just from the ability to learn and reason, but from the ability to strategically *manage* learning and reasoning. Its tripartite structure—a sub-symbolic substrate, a symbolic substrate, and a metacognitive controller—is designed to operationalize this management. Our goal is to present a blueprint for an AI that is verifiable by design, accountable in its operations, and possesses a rudimentary form of self-awareness about its own cognitive operations, thus addressing the deepest gap in the quest for artificial general intelligence.

2. A Critical Review of Architectures for Intelligence

To contextualize Cognito's contribution, it is essential to analyze the evolutionary lineage of AI architectures, from their foundations in cognitive science to contemporary attempts to organize LLM-based agents. This review reveals a recurring pattern: rapid progress in computational capabilities that outpaces the development of robust architectural principles to govern them.

2.1. Classic Cognitive Architectures: Foundations and Modern Relevance

The first attempts to create artificial general intelligence were deeply influenced by cognitive science, seeking to model the mechanisms of human cognition. Two of the most prominent architectures of this era, ACT-R and Soar, established fundamental principles that remain relevant.

ACT-R (Adaptive Control of Thought–Rational) was designed with the primary goal of quantitatively modeling human cognition. Its strength lies in its detailed, modular approach, which clearly distinguishes between declarative (facts) and procedural (production rules) memory systems. Crucially, ACT-R also incorporates sub-symbolic processes that govern the activation of knowledge chunks, allowing it to accurately model the nuances of human decision-making, including individual differences and varying strategies.

Soar, on the other hand, was developed with the goal of being an architecture for general intelligence, both human and artificial. Its focus is on problem-solving through the exploration of "problem spaces" and on learning through a unified mechanism called "chunking," which compiles successful problem-solving sequences into new rules. Unlike ACT-R, Soar adopts a more unified approach to agent data, treating all knowledge as production rules that operate on a working memory. Soar was conceived as a task-independent infrastructure, a blueprint for a general AI agent.

Despite their theoretical importance, these classic architectures face significant challenges in the modern era. Their implementation often requires complex and specialized coding, and their integration with large-scale deep learning architectures is non-trivial, which hinders their widespread adoption.

2.2. Language Agents and the CoALA Framework

The sudden advent and power of LLMs have led to a Cambrian explosion of "language

agents"—systems that use an LLM as their cognitive core to interact with environments and tools.[12] This rapid development has occurred in a largely ad-hoc manner, resulting in a proliferation of custom and inconsistent terminologies and abstractions, making it difficult to compare systems and discern clear architectural progress.[12]

The **CoALA (Cognitive Architectures for Language Agents)** framework emerged as a reactive attempt to impose order on this chaotic field.[13] Drawing inspiration from classic cognitive architectures, CoALA proposes a conceptual "blueprint" for organizing and designing language agents. It organizes agents along three main dimensions: (1) modular

memory components (short-term working memory and long-term memories such as episodic, semantic, and procedural); (2) a structured **action space** (internal actions to manipulate memory and external actions to interact with the environment); and (3) a generalized **decision-making process**, typically a plan-execute loop.[17]

CoALA represents a valuable conceptual bridge, contextualizing modern language agents within the broader history of AI. However, while it provides a useful taxonomy of *what* agents do (reason, retrieve, learn), its decision-making model lacks an explicit, self-regulating metacognitive layer. It describes a procedure, but not a governance mechanism that decides *how* and *why* that procedure should be self-consciously adapted in response to new information or internal uncertainty. The very existence of CoALA is proof that the field developed potent capabilities (LLMs) before developing the architectural principles to effectively control them.

2.3. The Neuro-Symbolic Landscape: A Taxonomy of Integration

The field of NeSy, in its quest to combine learning and reasoning, has developed several taxonomies to classify the integration patterns between neural and symbolic components. These taxonomies primarily focus on the structure of their interconnection. For example, a common taxonomy distinguishes between:

- **Symbolic[Neuro]**: Systems where an overarching symbolic solver uses neural subroutines for specific tasks, such as perception or statistical learning. The overall control remains on the symbolic side.
- **Neuro|Symbolic**: Systems organized as a pipeline, where a neural component first processes raw data (e.g., extracting features from an image) and passes a structured representation to a symbolic component for higher-level reasoning.

These classifications are structural and largely static. They describe the "wiring diagram" of the system but do not address the dimension of dynamic orchestration. Intelligent behavior in complex, unpredictable environments requires real-time adaptation of *how* these components

are used. For example, an agent must be able to decide whether a specific problem requires fast, intuitive pattern matching (neural) or a slow, deliberate logical proof (symbolic). This decision-making process is a higher-order, or *meta-level*, function that is not captured by existing structural taxonomies. This is precisely the gap that Cognito's Metacognitive Controller is intended to fill, moving from static wiring diagrams to dynamic cognitive governance.

2.4. A Comparative Taxonomy of AI Architectures

To synthesize this review and highlight Cognito's novelty, the following table presents a direct comparison between the discussed architectures. The comparison is structured to emphasize the evolution of approaches and to isolate Cognito's unique contribution in the domain of metacognitive control.

Feature	ACT-R	Soar	CoALA-based Agents	Standard NeSy Models	**Cognito**
Primary Goal	Human Cognitive Modeling	General Problem Solving	Systematization of LLM Agents	Bridging Learning and Reasoning	**Verifiable General Intelligence**
Memory Representation	Modular Declarative/ Procedural	Unified Problem Spaces	Ad-hoc Working/Long-Term	Hybrid	**Metacognitively-Managed Multi-Memory**
Reasoning Mechanism	Production Rules	Chunking	Chain-of-Thought (LLM)	Logic Solvers	**Bidirectional Neuro-Symbolic Translation**
Learning Process	Sub-symbolic Tuning	Chunking	Fine-tuning / RLHF	End-to-End Differentiation	**Metacognitively-Guided**

					Self-Correction
Explainability	Traceable Rules	Post-hoc	Logic-based Justification	**Intrinsic Verifiability**	
Metacognitive Control	None	None	Implicit/Procedural	None	**Explicit, First-Class Component**

Table 1: A taxonomic comparison of prominent AI architectures. The table highlights the evolution of goals and mechanisms, culminating in Cognito's introduction of explicit metacognitive control as a fundamental architectural component, a feature absent or merely implicit in prior approaches.

3. The Cognito Architecture: A Metacognitively-Governed System

The Cognito architecture is designed to operationalize metacognitive governance. Rather than being a linear pipeline or a static amalgam of components, Cognito is a dynamic, interconnected system whose global behavior is orchestrated by an executive control layer.

3.1. Foundational Principles: Holonomy and Self-Regulation

Cognito's design philosophy draws inspiration from two core principles. The first is drawn from **Holonomic Brain Theory**.[18] While we do not adopt this theory as a literal model of neuroscience, we extract its central principle: information is processed in a distributed, multi-scale, and parallel fashion across qualitatively different domains (in the brain, quantum and classical; in Cognito, sub-symbolic and symbolic). This motivates a design that is not a simple sequence of steps, but a dynamic, interconnected whole, where the whole is greater

than the sum of its parts.

The second and more important principle is **self-regulation**, which derives directly from the definition of metacognition. The architecture is explicitly designed to be introspective. It must be able to monitor its own internal states (such as confidence in a prediction), evaluate the consistency of its beliefs, and adapt its computational strategy accordingly. This capacity for self-regulation is what distinguishes Cognito from other architectures.

3.2. Architectural Components

Cognito is composed of four main components, each with a distinct function, but all under the governance of the Metacognitive Controller.

3.2.1. The Sub-Symbolic Substrate (System 1)

This layer is responsible for fast, intuitive, and massively parallel processing, analogous to System 1 thinking. Its function is to handle raw, high-dimensional data, recognize patterns, detect anomalies, and rapidly generate candidate hypotheses or "intuitions" that can then be rigorously examined by the symbolic layer.

- **Enabling Technology:** For this layer, the use of **Structured State Space Models (SSMs)**, such as Mamba, is proposed as an alternative to Transformers.[18] The justification for this choice is functional. System 1 thinking needs to be fast and capable of handling long-range dependencies to provide effective intuitions. Transformers, with their quadratic computational complexity with respect to sequence length, can become a bottleneck.[18] SSMs, on the other hand, offer linear or near-linear complexity and significantly faster inference, making them a superior architectural fit for the functional role of the sub-symbolic substrate in Cognito.[18]

3.2.2. The Symbolic Substrate (System 2)

This layer is the seat of slow, deliberate, and serial reasoning, analogous to System 2 thinking. It is responsible for logical deduction, causal reasoning, planning, formal verification, and the generation of explicit, interpretable explanations for its conclusions.

- **Enabling Technology:** The core of this layer is a formal **Domain-Specific Language (DSL)** and a **Neuro-Symbolic Program Synthesizer**.[19] This synthesizer is a hybrid system that uses a neural network to guide the search in the space of possible programs within the DSL. Given a specification—which can consist of input-output examples, logical constraints, or natural language descriptions—the synthesizer generates a program in the DSL that satisfies that specification. The crucial advantage of this approach is that the output is not an opaque network weight, but an explicit, structured piece of code that can be analyzed, interpreted, and, most importantly, formally verified.[5]

3.2.3. The Bidirectional Neuro-Symbolic Translation Bus

This component functions as the critical interface between the two substrates. Its function is to perform the challenging but essential task of translating between the continuous, high-dimensional representations of the sub-symbolic substrate (feature vectors, embeddings) and the discrete, compositional structures of the symbolic substrate (logical formulas, programs). This process is often referred to as "cognitive abstraction".[8] The translation is bidirectional: neural outputs (e.g., "this image likely contains a cat and a rug") are abstracted into symbols that logical reasoning can manipulate, and symbolic goals or constraints (e.g., "find a plan to move the cup to the table without spilling it") are translated into loss functions or heuristics that can guide the sub-symbolic substrate.

3.2.4. The Metacognitive Controller (The "Self")

This is Cognito's central innovation and its governing component. The Metacognitive Controller (MCC) does not perform the primary computation; instead, it orchestrates the other components. Its key executive tasks include:

1. **Problem Triage:** Upon receiving a new task, the MCC determines the optimal computational strategy. Is this a pattern recognition problem best suited for the sub-symbolic substrate? Does it require a rigorous proof from the symbolic substrate? Or does it need a hybrid approach where System 1 generates candidates and System 2 verifies them?
2. **Confidence Monitoring:** The MCC continuously evaluates the uncertainty or confidence level of the outputs from both substrates. If the sub-symbolic layer produces a low-confidence prediction, the MCC can trigger the symbolic layer to perform a deeper, more rigorous logical analysis.
3. **Logical Consistency Enforcement:** This is a critical function where concepts like **Logic**

Tensor Networks (LTNs) are repurposed for real-time verification. The MCC maintains a set of inviolable logical axioms (background knowledge, e.g., $\forall x(\text{Bird}(x) \rightarrow \text{HasWings}(x))$). At any moment, the MCC can take the current state of the system (its beliefs or outputs), represent it as tensors, and use the structure of an LTN to calculate the degree of satisfaction of these axioms. If the satisfaction level falls below a threshold—indicating a logical contradiction—the MCC can flag an error and initiate a correction process, forcing a re-evaluation. This transforms LTNs from a static training constraint into a dynamic "sanity checker" or "conscience" for the entire system.

4. **Learning Regulation:** The MCC actively guides the learning process. Instead of a blind optimization of a loss function, the MCC can identify specific knowledge gaps or logical inconsistencies and direct the system to actively seek new data or perform internal reasoning (e.g., synthesize new rules) to resolve these deficiencies. This constitutes a form of active, self-directed learning.

3.3. Functional Specification of the Cognito Architecture

The following table provides a structured summary of Cognito's components, detailing their functions, enabling technologies, and interactions, to offer a clear blueprint of the architecture.

Component & Analogy	Key Functions	Enabling Technologies	Input	Output	Interaction with Controller
Sub-Symbolic Substrate (System 1 / Intuition)	Fast pattern matching, hypothesis generation, perception	State Space Models (Mamba), Graph Neural Networks	Raw sensory data, vector embeddings	Candidate classifications, anomaly scores, latent representations	Reports confidence scores, receives attention directives
Symbolic Substrate (System 2 / Reason)	Logical reasoning, planning, formal verification, explanation	Neuro-Symbolic Program Synthesizer, Dependently Typed	Formal specifications, logical constraints, I/O examples	Verifiable programs, logical proofs, action plans	Receives problems for formal solving, reports proofs/failu

	generation	Languages			res
Bidirectional Translation Bus (Interpreter)	Concept abstraction, symbol grounding	Attention models, structured embedding learning	Latent representations, symbolic programs	Grounded symbols, semantic loss functions	Orchestrated by controller to mediate communication
Metacognitive Controller (The "Self" / Executive)	Problem triage, confidence monitoring, consistency enforcement, learning regulation	Logic Tensor Networks (for verification), Hierarchical Reinforcement Learning	Internal states (confidence, consistency), task goals	Cognitive resource allocation, solving strategies, correction requests	**Governs all other components**

Table 2: A functional specification of the Cognito architecture. The table details the role of each component, the underlying technologies, and, crucially, their interactions with the Metacognitive Controller, illustrating how central governance is achieved.

4. Verifiability and Accountability by Design

One of the most significant promises of the Cognito architecture is its ability to shift the paradigm of Explainable AI (XAI) and Responsible AI (RAI). Instead of treating explainability and safety as features to be added post-hoc, Cognito integrates them into its very architectural fabric, enabling verifiability by design.

4.1. From Post-Hoc Explainability to Intrinsic Verifiability

The prevailing approach to XAI focuses on post-hoc methods, such as LIME and SHAP, which attempt to approximate or explain the decisions of a black-box model after they have been

made.[3] While useful for debugging and gaining some intuition, these techniques are fundamentally insufficient for safety-critical systems. Their explanations are approximations and offer no guarantees about the model's behavior in all possible scenarios.

Cognito, in contrast, is designed for **intrinsic verifiability**. Explainability is not an external layer but an emergent property of its symbolic substrate. When the symbolic substrate solves a problem, the output is a program. This program is not an explanation *about* the reasoning; it *is* the reasoning, captured in an explicit, human-readable, and formally analyzable form. This represents a fundamental shift from approximate explanations to exact, verifiable justifications.

4.2. Formal Verification through Dependently Typed Programming

The strength of Cognito's approach lies in the combination of neuro-symbolic program synthesis with the power of dependently typed programming languages, such as Idris, Agda, or F*. This combination creates an end-to-end verification chain.

- **Mechanism:** Dependently typed languages have extraordinarily expressive type systems that allow a program's properties to be encoded directly within its types. For example, the signature of an array access function can have a type that requires the index to be mathematically proven to be within the array's bounds: val array_access : a:array int -> i:nat{i < length a} -> int. A program written in such a language will only compile if a proof of the correctness of these properties can be constructed, often with the assistance of an SMT (Satisfiability Modulo Theories) solver. The type becomes a formal specification, and successful compilation becomes an act of formal verification.
- **Application in Cognito:** The process works as follows:
    1. The sub-symbolic substrate and the neural program synthesizer *propose* a program in the DSL as a candidate solution for a task.
    2. This program is then submitted to a type checker of a dependently typed language.
    3. Safety, fairness, and robustness constraints for the task have been pre-encoded as dependent types. For example, a planning algorithm for a robot might have a type that proves it will never enter a state defined as unsafe, or that a resource allocation decision satisfies a pre-defined fairness metric.
    4. The proposed program is only accepted and executed if it is "well-typed"—that is, if the type checker can prove that it satisfies all the safety and correctness specifications encoded in its types. Otherwise, it is rejected, and the Metacognitive Controller can instruct the synthesizer to find an alternative solution.

This process elevates verification from the status of a post-development testing activity to a design-time integrated guarantee, providing a level of safety that purely neural systems

cannot achieve.

4.3. A Framework for Responsible AI (RAI)

This intrinsic verification capability provides a solid foundation for achieving the goals of Responsible AI (RAI).[7]

- **Transparency:** The generated symbolic programs are inherently transparent. They provide a complete, logical, and human-readable explanation of the reasoning process that led to a decision, directly addressing the black-box problem.[7]
- **Fairness:** Fairness constraints, such as demographic parity or equality of opportunity, can be formalized as logical formulas or type properties. The Cognito architecture can then formally verify that its decision-making behavior adheres to these constraints, instead of relying solely on mitigating bias in the training data.[7]
- **Robustness:** By formally verifying that the system's behavior remains within safe bounds, Cognito can be robust against the catastrophic failures and unpredictable behavior that can afflict purely neural systems when faced with unexpected or adversarial inputs. Safety guarantees can be encoded and enforced, ensuring predictable and bounded behavior in critical applications.[7]

In short, Cognito does not treat responsibility as a desirable goal, but as a verifiable architectural property.

5. Discussion: Implications and Future Trajectories

The introduction of the Cognito architecture has implications that extend beyond a simple technical advancement. It proposes a fundamental reorientation in how we conceive and build intelligent systems, with potential impacts on the quest for AGI, research challenges, and the future of AI infrastructure.

5.1. Cognito vs. The State of the Art

Recapitulating the comparative analysis presented in Table 1, Cognito's key differentiator from

all previous approaches—from classic cognitive architectures to the latest neuro-symbolic models—is the explicit modeling and operationalization of metacognition. While other architectures may have implicit control mechanisms or feedback loops, Cognito is the first to posit a Metacognitive Controller as a first-class component, with the explicit responsibility of governing the system's cognitive resources. This shift from an architecture of "workers" (learning and reasoning components) to an architecture with "management" (the controller) is its main conceptual leap. It directly addresses the 5% gap in metacognition research, transforming it from a neglected area into an architectural cornerstone.

5.2. Metacognition as a Catalyst for AGI

The quest for Artificial General Intelligence (AGI) is the ultimate goal of much of AI research. We argue that metacognitive capabilities, such as those embodied in Cognito, are not just desirable features of intelligence, but prerequisites for the general and adaptive intelligence that AGI seeks to achieve.[18] A true AGI must be able to operate in open and unknown domains. This requires the ability to:

- **Recognize ignorance:** Knowing what one does not know is fundamental to seeking new knowledge. The Metacognitive Controller, through its confidence monitoring, operationalizes this capability.
- **Adapt strategy:** Intelligence is not the application of a single powerful algorithm, but the selection of the right cognitive tool for the job. The MCC's problem triage function models this strategic resource allocation.
- Self-correction: Learning from mistakes requires the ability to detect them in the first place. The MCC's enforcement of logical consistency provides a mechanism to detect and initiate the correction of contradictory beliefs.
  By focusing on the management of cognitive processes, Cognito offers a more plausible path to the robust autonomy and adaptability that define AGI.

5.3. Challenges and Open Problems

Despite its potential, the realization of the Cognito architecture faces significant theoretical and engineering challenges that represent rich areas for future research.

- **The Translation Problem:** Bidirectional neuro-symbolic translation remains one of the most difficult and open problems in AI. How to effectively translate the rich, sub-symbolic vector representations into crisp symbolic structures without loss of information, and vice versa, is a fundamental research question.[22]

- **Scalability of Verification:** Formal verification, especially with dependent types, is computationally intensive. While feasible for critical programs or modules, applying it to all outputs of the symbolic substrate in real-time may not be tractable. Research into more scalable verification techniques or selective verification guided by the MCC will be crucial.[5]
- **Defining Metacognitive Rules:** The "rules" that govern the Metacognitive Controller itself need to be defined. How does the system learn to become a better manager of its own cognitive resources? This points to approaches like hierarchical reinforcement learning, where the MCC learns a high-level policy to orchestrate the low-level policies of the substrates.

5.4. A Decentralized Future for Cognito

Finally, Cognito's architectural principles—modularity, separation of concerns, and verifiable components—make it exceptionally well-suited for the emerging trends in decentralized AI infrastructure. Instead of existing as a monolithic model in a centralized data center, Cognito can be implemented as a distributed, collaborative process.

Systems like **Gradient Network's Parallax and Lattica** are being developed to enable precisely this future. Parallax is designed as a "world inference engine" that decomposes models and executes their parts across a global mesh of heterogeneous hardware, from data center GPUs to consumer devices.[10] This "Swarm" architecture [23] aligns perfectly with Cognito's modular structure. One can envision a future where Cognito's computationally intensive sub-symbolic tasks run on a swarm of GPUs, the more logic-intensive symbolic verification tasks run on CPUs, and the Metacognitive Controller orchestrates this distributed workflow through a peer-to-peer communication protocol like Lattica.[24] This approach not only democratizes access to powerful AGI, promoting "intelligence sovereignty" [23], but also increases the system's resilience and robustness, mirroring the holonomic principle of distributed processing.

6. Conclusion

This paper has addressed a fundamental tension at the heart of contemporary artificial intelligence: the gap between the pattern recognition proficiency of deep learning models and the demands of robustness, explainability, and systematic reasoning of genuine cognition. Although Neuro-Symbolic AI has emerged as the primary path to resolve this tension, we have

identified a critical and quantified flaw in its current research agenda: a profound neglect of metacognition.

We have introduced Cognito, a novel architecture designed to fill this void. The central contributions of this work are threefold:

1. **A New Architecture:** Cognito is the first AI architecture to explicitly model metacognition as a central governing component. Its Metacognitive Controller introduces an executive management layer that monitors, evaluates, and dynamically regulates its sub-symbolic and symbolic substrates, enabling strategic allocation of cognitive resources and principled self-regulation.
2. **A Path to Verifiability:** We have proposed a concrete methodology for building intrinsically verifiable AI systems. By combining neuro-symbolic program synthesis with the verification power of dependently typed programming languages, Cognito transforms safety, fairness, and robustness from post-hoc tested properties into design-time proven guarantees.
3. **A Foundation for Trustworthy AGI:** Ultimately, Cognito offers a principled framework for developing the next generation of AI. By integrating metacognitive governance and formal verifiability at its core, it establishes a path toward systems that are not only more capable, but also more robust, transparent, and aligned with human values.

The challenges ahead in realizing such a system are immense, but the trajectory is clear. The journey toward artificial general intelligence will not be achieved merely by scaling larger models, but by designing architectures that are smarter, more reflective, and, ultimately, more self-aware.

Referências citadas

1. Formal Methods - Electrical and Computer Engineering, acessado em outubro 2, 2025, https://users.ece.cmu.edu/~koopman/des_s99/formal_methods/
2. In a dependently typed programming language is Type-in-Type practical for programming? - Stack Overflow, acessado em outubro 2, 2025, https://stackoverflow.com/questions/61930740/in-a-dependently-typed-programming-language-is-type-in-type-practical-for-progra
3. [PLDI'25] Partial Evaluation, Whole-Program Compilation - YouTube, acessado em outubro 2, 2025, https://www.youtube.com/watch?v=S3ByUa1L5TM
4. Cognitive Architectures for Language Agents - arXiv, acessado em outubro 2, 2025, https://arxiv.org/pdf/2309.02427
5. Neurosymbolic AI: Bridging Neural Networks and Symbolic ..., acessado em outubro 2, 2025, https://www.netguru.com/blog/neurosymbolic-ai
6. Toward Transparent Modeling: A Scoping Review of Explainability for Arabic Sentiment Analysis - MDPI, acessado em outubro 2, 2025, https://www.mdpi.com/2076-3417/15/19/10659
7. (PDF) Towards Responsible AI through NeuroSymbolic Integration ..., acessado em outubro 2, 2025,

https://www.researchgate.net/publication/395129655_Towards_Responsible_AI_through_NeuroSymbolic_Integration_A_Survey
8. Cognitive Architectures for Language Agents - Princeton University, acessado em outubro 2, 2025, https://collaborate.princeton.edu/en/publications/cognitive-architectures-for-language-agents
9. Neural-guided, Bidirectional Program Search for Abstraction and Reasoning - arXiv, acessado em outubro 2, 2025, https://arxiv.org/abs/2110.11536
10. (PDF) Parallax - A New Operating System Prototype Demonstrating ..., acessado em outubro 2, 2025, https://www.researchgate.net/publication/221015688_Parallax_-_A_New_Operating_System_Prototype_Demonstrating_Service_Scaling_and_Service_Self-Repair_in_Multi-core_Servers
11. Symbolic, Distributed, and Distributional Representations ... - Frontiers, acessado em outubro 2, 2025, https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2019.00153/full
12. List of P2P protocols - Wikipedia, acessado em outubro 2, 2025, https://en.wikipedia.org/wiki/List_of_P2P_protocols
13. Cognitive architecture - Wikipedia, acessado em outubro 2, 2025, https://en.wikipedia.org/wiki/Cognitive_architecture
14. Introduction to the Soar Cognitive Architecture, acessado em outubro 2, 2025, https://acs.ist.psu.edu/misc/nottingham/pst2/hungry-thirsty/full.html
15. Evaluating Neuro-Symbolic AI Architectures: Design Principles, Qualitative Benchmark, Comparative Analysis and Results - OpenReview, acessado em outubro 2, 2025, https://openreview.net/pdf?id=yCwcRijfXz
16. [PDF] Cognitive Architectures for Language Agents - Semantic Scholar, acessado em outubro 2, 2025, https://www.semanticscholar.org/paper/e4bb1b1f97711a7634bf4bff72c56891be2222e6
17. Parallax | Gradient, acessado em outubro 2, 2025, https://docs.gradient.network/research/the-gradient-stack/parallax
18. Q&A: Can Neuro-Symbolic AI Solve AI's Weaknesses? - TDWI, acessado em outubro 2, 2025, https://tdwi.org/articles/2024/04/08/adv-all-can-neuro-symbolic-ai-solve-ai-weaknesses.aspx
19. Cognitive architectures: Research issues and challenges - GitHub, acessado em outubro 2, 2025, https://raw.githubusercontent.com/SoarGroup/website-downloads/main/pubs/cogarch.cogsys08.pdf
20. CoALA (Cognitive Architectures for Language Agents) - Google Groups, acessado em outubro 2, 2025, https://groups.google.com/g/rssc-list/c/3H_tPyFaLfw
21. Vector-space models of semantic representation from a cognitive perspective: A discussion of common misconceptions, acessado em outubro 2, 2025,

https://boa.unimib.it/retrieve/handle/10281/261023/379692/GuentherRinaldiMarelli_inpress.pdf
22. Navigating Key AI Translation Challenges | Galileo, acessado em outubro 2, 2025, https://galileo.ai/blog/ai-translation-challenges
23. Neuro-Symbolic AI in 2024: A Systematic Review - arXiv, acessado em outubro 2, 2025, https://arxiv.org/html/2501.05435v1
24. Neuro-Symbolic AI: Let's go back to the start | by Eric Papenhausen | Medium, acessado em outubro 2, 2025, https://medium.com/@epapenha_40736/neuro-symbolic-ai-lets-go-back-to-the-start-cca9be15002