Cognitive LLMs: Toward Human-Like Artificial Intelligence by Integrating Cognitive Architectures and Large Language Models for Manufacturing Decision-making

Siyu Wu [a,*], Alessandro Oltramari [b,c], Jonathan Francis [b,c], C. Lee Giles [a] and Frank E. Ritter [a]
[a] *College of Information Sciences and Technology, The Pennsylvania State University, University Park, USA*
E-mail: {sfw5621, clg20, frank.ritter}@psu.edu
[b] *Bosch Center for Artificial Intelligence, Pittsburgh, USA*
[c] *Carnegie Mellon University, Pittsburgh, USA*
E-mail: {alessandro.ultramarine, jon.francis}@us.bosch.com

Abstract. Resolving the dichotomy between the human-like yet constrained reasoning processes of Cognitive Architectures and the broad but often noisy inference behavior of Large Language Models (LLMs) remains a challenging but exciting pursuit, for enabling reliable machine reasoning capabilities in production systems. Because Cognitive Architectures are famously developed for the purpose of modeling the internal mechanisms of human cognitive decision-making at a computational level, new investigations consider the goal of informing LLMs with the knowledge necessary for replicating such processes, e.g., guided perception, memory, goal-setting, and action. Previous approaches that use LLMs for grounded decision-making struggle with complex reasoning tasks that require slower, deliberate cognition over fast and intuitive inference—reporting issues related to the lack of sufficient grounding, as in hallucination. To resolve these challenges, we introduce `LLM-ACTR`, a novel neuro-symbolic architecture that provides human-aligned and versatile decision-making by integrating the ACT-R Cognitive Architecture with LLMs. Our framework extracts and embeds knowledge of ACT-R's internal decision-making process as latent neural representations, injects this information into trainable LLM adapter layers, and fine-tunes the LLMs for downstream prediction. Our experiments on novel Design for Manufacturing tasks show both improved task performance as well as improved grounded decision-making capability of our approach, compared to LLM-only baselines that leverage chain-of-thought reasoning strategies. We release the code and data samples from our approach at https://github.com/SiyuWu528/LLM-ACTR.

Keywords: Cognitive architectures, Large language models

Introduction

Large-capacity neural foundation models, such as Large Language Models (LLMs), have gained considerable popularity for a wide range of prediction and decision-making tasks, spanning applications, such as robotics and control,

*Corresponding author. E-mail: sfw5621@psu.edu.

neural question-answering, scene understanding, code generation, mathematical reasoning. LLMs are trained on massive datasets, can be used both as discriminative scoring functions as well as generative models, and their capacity allows them to accumulate and retain vast amounts of knowledge [7, 16, 21, 26, 36]. On the surface, typical usage of LLMs mirrors '*system-1 reasoning processes*' [32, 73], from the dual-process theory of human cognition [40, 85], which provide fast, intuitive, and automatic reasoning—underpinning tasks like navigating daily environments and making quick decisions. Advancements in multi-agent LLM frameworks as well as emergent capabilities such as in-context learning [20, 21] have enabled LLMs to employ more sophisticated reasoning strategies, such as 'chain-of-thought' reasoning (CoT) [11]. These capabilities facilitate LLMs' pursuit of '*system-2 processes*' [81], which involve slower, deliberate cognition and critical thinking for complex tasks [16, 86]—essential for decision-making in realistic settings. While LLMs have shown promise in this area, key concerns remain, e.g., over discrepancies between LLM inference behavior and human reasoning [12, 50], in analyses showing that LLMs prioritize fast and intuitive "system-1" thinking over slower and deliberate analysis [32], and over issues of insufficient grounding such as hallucination [17]. These issues raise potential concerns about deployment settings where LLMs are left to perform inference, without having been first grounded on reliable knowledge sources or decision processes [93].

To alleviate these issues, we propose `LLM-ACTR`, which shows improved decision-making capabilities over LLMs by integrating intermediate representations extracted from a well-establish neuro-symbolic system: the ACT-R cognitive architecture [5, 67]. ACT-R offers an integrated theory of the mind — encompassing perception, memory, goal-setting, and action — and has been pivotal in developing synthetic agents for learning and training [6]. The representation extracted from ACT-R cognitive models serves as domain knowledge, infusing LLMs with decision-making augmentation. `LLM-ACTR` uses ACT-R models to represent human repeated decision-making with learning.

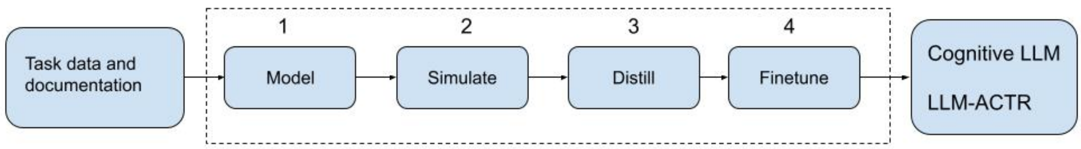

Fig. 1. Decision augmentation using a neural-symbolic cognitive architecture approach. (1) Tasks are modeled with cognitive architecture. (2) Cognitive model used to run stochastic simulation of task at scale. (3) Synthetic data are distilled from simulation and combined with prompt requests. (4) A fine tuning pipeline is used to calibrate open source LLM to perform decision augmentation for task in exercise.

We infuse ACT-R model's intermediate representations with the last hidden layers of open source LLM, and add a top classification layer for fine-tuning. The architecture is then deployed in unseen decision-making tasks. The LlaMa model family [80] was selected for this research, due to its full accessibility to network architecture, including its pre-trained weights, and its proven efficacy in prior applications involving the extraction of the last hidden layer for predicting behavior discrepancies [13]. This approach integrates the ACT-R model's representation of human-like decision-making patterns into the LLM, enhancing its ability to make decisions that are both human-aligned and explainable. The fine-tuned LLM transcends mere prediction of human decisions for unseen problems. Significantly, it outlines a road-map for enabling high-level machine reasoning through cognitive neuro-symbolic systems [61]. `LLM-ACTR` leverages the strengths of both LLMs and CAs by using LLMs' natural language processing and generative capabilities, complemented by the human-aligned reasoning and explainability offered by CAs.

This paper presents a deployment case of `LLM-ACTR` in manufacturing decision-making, demonstrating how this approach addresses the typically noisy inference behavior associated with off-the-shelf LLMs in real-world settings. The task is associated with the key aspect of Design For Manufacturing (DFM): enhancing product development and optimizing production system performance by improving time efficiency and reducing headcount costs [82]. In this work, we introduce Cognitive LLMs, describe our proposed framework `LLM-ACTR`, and provide experimental comparisons on a novel task with strong baselines.

Related Work

This section begins with introducing the state-of-the-art in cognitive psychology walking into LLMs Lab, decision intelligence in manufacturing, and cognitive decision-making. It then highlights the domain limitations of these approaches. Finally, the discussion moves to the current integration of cognitive architectures (CAs) and large language models (LLMs) to develop a more robust, unified theory of decision-making models.

Relating Cognitive Psychology to Human-Like Artificial Intelligence

Human-like artificial intelligence (HLAI), aimed at developing entities that possess capabilities similar to those of humans, has been a goal since the emergence of machines [55]. In recent years, the development of transformer-based large language models (LLMs) has revolutionized HLAI. These models, as representatives of HLAI artifacts [25], have demonstrated impressive human-level capabilities. We might be experiencing one of the most transformative revolutions in artificial intelligence. The influence of these advanced language models reaches beyond their original purposes, affecting multiple fields, including education [57], healthcare [79], and the job market [24]

Although these models sometimes display human-like behavioral traits, this is not consistently true. Analyzing the areas where LLMs currently fall short in replicating human cognition and behavior highlights the problems in exhibiting human-level capabilities that are unhuman-like [22], including behavior discrepancies between LLM inference behavior and human reasoning [12, 50], insufficient grounding [17], and hallucination [93].

The challenges mentioned have catalyzed a deeper integration of cognitive psychology with LLMs, toward not only human-level but also human-like artificial intelligence. Recent studies have brought cognitive psychology into the realm of LLMs, using cognitive psychology experiments to investigate and comprehend behaviors in these models, focusing more on behavioral insights than on conventional performance metrics [13, 19]. In addition, the use of LLMs' neural representations in behavioral psychological science research, which involves connections with prompt engineering, feature extraction, and fine-tuning, includes the following approaches [38]:

Feature Extraction. LLMs are used for feature extraction in psychological experiments. The process begins with passing text that mirrors a psychological experiment through the open-source LLM to capture contextualized embeddings from the final layer [13]. These embeddings can employed in various psychological experiments applications, such as predicting similarities between personality constructs [2], choices in reinforcement learning [12], or perceptions related to risk or health [91]. For tasks that require sequence prediction, decoder models are preferred due to their larger size and more extensive training data [38].

Zero-shot and Few-shot Learning. LLMs can generate results with little or no supervision through a technique called zero-shot learning. This approach enables the creation of categorical or numerical predictions, such as evaluating sentiments on social media [23], without requiring training specific to the task. Few-shot learning extends this concept by adding minimal supervision, such as a small number of example pairs, to improve the accuracy of the model.

Fine-Tuning. Fine-tuning smaller LLMs for specific tasks can achieve performance that matches or exceeds that of larger models under zero- or few-shot learning conditions. This involves adjusting model weights to improve task-specific outcomes.

Zero-shot and few-shot learning are often used alongside feature extraction and fine-tuning. For instance, one study uses embeddings from LlaMa in zero-shot learning to predict reinforcement learning outcomes from past behavioral studies [13]. However, similar research endeavors face significant challenges due to the high costs and extensive effort required to collect and expand large cognitive psychological datasets.

Common Model of Cognition, Cognitive Architectures, and Cognitive Model

To address the challenge of data scarcity, we introduce a suite of tools rooted in the Common Model of Cognition (CMC). CMC is a theoretical framework that presents a model of human cognition codified as a computational architecture [46]. The CMC is a brain-inspired framework validated by large-scale neuroscience data [74]. The CMC identifies core components and processes fundamental to human cognition, including memory, perception, motor actions, and decision-making. The model assumes a cyclical process where these components interact to produce intelligent behavior. The CMC includes a feature-based declarative long-term memory, a buffer-based working memory, a system for pattern-directed action invocation stored in procedural memory, and specialized systems for perception and action.

The CMC integrates essential features from various cognitive architectures [5, 45], which are computational frameworks designed to capture the invariant mechanisms of human cognition. These mechanisms include functions related to attention, control, learning, memory, adaptivity, perception, and action. Cognitive architectures propose a set of fixed mechanisms to model human behavior, functioning akin to agents and aiming for a unified representation of the mind. By using task-specific knowledge, these architectures not only simulate but also explain behavior through direct examination and real-time reasoning tracing. Two representative cognitive architectures are ACT-R and Soar.

ACT-R

ACT-R is a cognitive architecture and a theory of simulating and understanding human cognition [5]. Its theory is embodied in the ACT-R software, through which we can construct models that can store, retrieve, and process knowledge, as well as explain and predict performance [15]. There are currently two kinds of knowledge representations in ACT-R, and they are declarative knowledge and procedural knowledge. Declarative knowledge consists of chunks of memory (e.g., apple is a kind of fruit), while procedural knowledge performs basic operations, moves data among buffers, and identifies the next instructions to be executed (e.g., to submit your answer, you have to click the submit bottom). When the model is driving a bus in a first-person perspective, these pieces of information will contain information such as what visual items to look at and what tasks to do next.

Soar

Soar is a general cognitive architecture that provides a computational infrastructure that resembles the cognitive capabilities exhibited by a human. Soar implements knowledge-intensive reasoning that enables execution of rules based on the context. It also has the capability to integrate learning into the intelligent agent using chunking or reinforcement learning. Soar has its origins in the groundbreaking work done by Newell and Simon around the 1950s through the mid-1970s, also inspired by the ”General Problem Solver” created by Ernst and Newell. While ACT-R was designed to model human behavior, Soar was inspired by it. Current understanding and hypotheses regarding cognitive architecture are incorporated into Soar 9, which has been in development for over 30 years and continues to evolve gradually. Soar's general computing concept is based on: objectives, problem spaces, states and operators [45, 60]. Soar encompasses multiple memory constructs (e.g., semantic, episodic, etc.) and learning mechanisms (e.g., reinforcement, chunking etc.) and is a programmable architecture with an embedded theory. This enables executing Soar models on embedded system platforms and studying the design problem through rapid prototyping and simulation.

Decision Intelligence in Manufacturing

Industry 4.0 aims to create 'intelligent factories,' where advanced manufacturing technologies facilitate smart decision-making through real-time communication and cooperation among humans, machines, and sensors [34]. One example of this is smart scheduling, which employs advanced models and algorithms using sensor data [70].

Decision intelligence [48] is a crucial component of smart scheduling and comprises three stages. **Decision support**. Machines provide basic tools to aid human decision-making, such as alerts, analytics, and data exploration. Here, the decisions are made entirely by humans. **Decision augmentation.** Machines take on a more proactive role in the decision-making process. They analyze data and generate recommendations and predictions for decision-makers to review and validate. Humans can base their decisions on these suggestions, or they can collaborate with the machine to refine the recommendations. **Decision Automation.** Machines handle both the decision-making and execution steps autonomously. Humans maintain a high-level overview, monitoring risks and unusual activities, and regularly review outcomes to enhance the system.

A value stream map (VSM) is a critical tool in realizing decision intelligence, functioning as an advanced flowchart that visualizes and controls the production line [53]. VSM meticulously tracks metrics such as inputs, outputs, processes, overall equipment effectiveness (OEE), and cycle times—all vital for analyzing quality and efficiency in production control. However, plant managers encounter significant challenges when transitioning VSM in production management from decision support to decision augmentation. These challenges stem from the difficulty of applying VSM concepts to complex, real-world scenarios characterized by numerous intertwined variables.

Cognitive Decision Making

The ACT-R architecture has been widely applied to build models that automate decision-making tasks across psychology and computer science. The modeling approaches used include: (a) strategy or rule-based, where different problem-solving strategies are implemented through various production rules and successful strategies are rewarded [10, 89]; (b) exemplar or instance-based, which relies on past experiences stored in declarative memory to solve problems [30]; and (c) hybrid approaches that combine strategies and exemplars [65].

ACT-R was chosen for this study to provide the intermediate representations of real time reasoning steps. Three key features distinguish the use of ACT-R in creating models for decision-making tasks that involve learning: **Self-configuration**: ACT-R efficiently translates instructions into structured rules, forming the basis for task-specific production rules that enhance the efficiency of task execution. **Modular design mirroring human cognition**: ACT-R's modules emulate human cognitive functions: perceptual modules update the system's view of the environment, a goal module tracks progress towards objectives, a declarative module uses past experiences for contextual understanding, and a central buffer system enables communication between modules. Additionally, the central production system recognizes patterns to initiate coordinated actions. **Subsymbolic processes for decision-making**: ACT-R excels in its ability to reliably retrieve relevant memories and activate appropriate rules, ensuring both efficient and adaptive performance in decision-making tasks, such as skills training. It does so at a pace that mirrors human performance and offers the opportunity to model learning during this process.

However, ACT-R models do not generally accept natural language as input and cannot easily or routinely generalize across different tasks, even within the same domain, which limits its flexibility for decision-making. In contrast, `LLM-ACTR` combines the strengths of both LLMs and ACT-R models by leveraging the natural language processing and generative capabilities of LLMs, and making decisions that are grounded by those of ACT-R models.

Integration of Cognitive Architectures and LLMs

CAs face limitations due to domain restrictions, which have hampered their broader application. They are incapable of processing natural language, they are limited to areas that can be described by logical rules, and they require a significant number of pre-defined rules to function. Intriguingly, LLMs [16] offer potential solutions to mitigating these weaknesses. LLMs can process a variety of text inputs and are more flexible than rule-based systems. Additionally, they seem to learn rules implicitly, through pre-training, eliminating the need for manual rule creation. Hence, the notion of integrating CAs and LLMs is attractive, for leveraging the strengths of both approaches and thereby creating a more robust unified theory of computational models. This integration can take several forms, however, e.g., leveraging the implicit world knowledge of LLMs to replace the CAs' declarative knowledge mechanisms or to enhance their traditional symbolic mechanisms for procedural knowledge [42, 88]. Additional research explores how principles from cognitive architectures can guide the design of LLM-based agent frameworks [75], demonstrating a comprehensive integration effort that spans from knowledge representation to interaction with the environment. However, to our knowledge, unlike these previous efforts that incorporate LLMs into CAs, there is currently no research focusing on assimilating the advantages of CAs into LLMs. In this paper, we leverage a cognitive architecture to ground the reasoning process and outputs of LLMs; by assimilating a neural representation of ACT-R model within LLMs, we aim to enhance LLMs' human alignment and explainability.

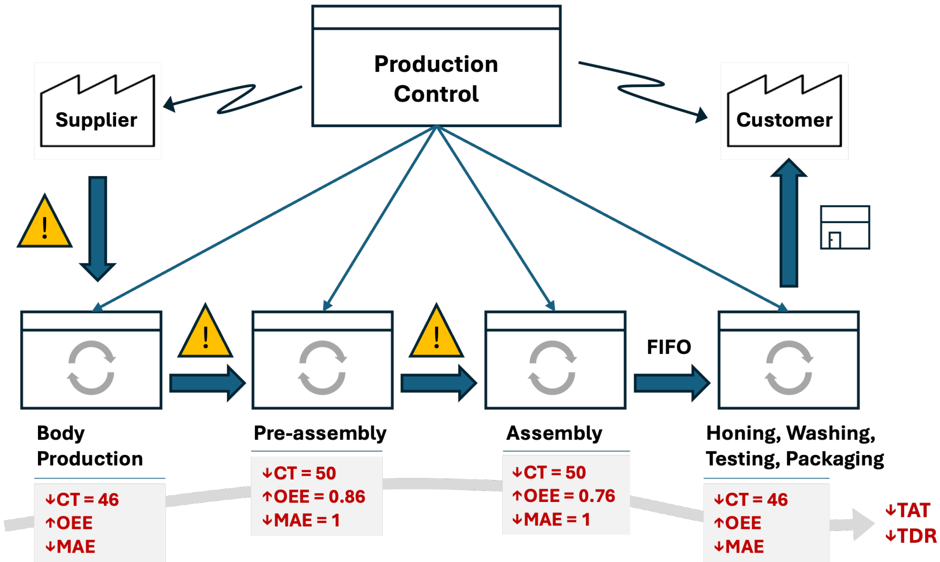

Fig. 2. A *Value Stream Map* of our manufacturing task process.

Problem Definition: Design for Manufacturing

We define the terminology that constitutes our problem. The problem setting is a prototypical manufacturing production-line workflow, from supplier to customer, for which there exists a Value Stream Map (VSM; see Figure 2), which allows for tracking the efficiency at different sectors of the process and abstracts the overall problem for mathematical modeling and optimization. Key sectors include: Body Production, Pre-Assembly, Assembly, Honing, Washing, Testing, and Packaging. Early sectors pose potential efficiency problems in the workflow and may warrant optimization (triangles), while later stages are governed by *First-In-First-Out* (FIFO) processes. The metrics at each stage include Cycle Time (CT), Overall Equipment Effectiveness (OEE), and Mean Absolute Error (MAE); the flow progresses through each stage, aiming for efficient operation, performance monitoring, and error minimization to ensure high-quality production output and timely customer delivery.

Focused on maintaining stable output for manufacturing plants, we consider plant managers' feedback alongside the VSM structure to define two decision-making problems that aim to reduce Total Assembly Time (TAT) while minimizing Total Defect Rate (TDR). An agent $\mathcal{G}$ is a predictive model that takes a natural language question $\mathcal{Q}$ as a prompt, along with N snapshots of the sector-wise production flow data {CT, OEE, MAE}. In a *single-facet decision-making problem*, $\mathcal{G}$ outputs a binary decision (0 or 1) on which of two sectors, pre-assembly or assembly, requires a time reduction. In a more-challenging *multi-faceted decision-making problem*, $\mathcal{G}$ should output the same binary decision as before, about which sector should be the optimization target, along with an optimization *strategy* S. Here, S is a strategy defined by one of several decision-making personas that govern manufacturing process management, which we refer to in the manuscript as 'novice', 'intermediate', and 'expert'.

Cognitive LLMs: Hybrid Architectures for Human-Aligned Decision Making

We start by providing a brief background on the central components of the ACT-R Cognitive model, before providing details about our proposed Cognitive LLM framework, `LLM-ACTR`. Our approach demonstrates essential characteristics, derived from ACT-R, which are crucial for augmenting decision-making using foundation models with cognitive reasoning.

Cognitive Architectures based on ACT-R

LLM-ACTR relies on an ACT-R cognitive model capable of (1) executing tasks from problem definition using decision-making behaviors observed in humans and retrieving knowledge representations similarly, (2) integrating personas ranging from novice to intermediate and expert levels, and (3) simulating the reinforcement learning processes of decision-makers as they transition from novice to expert.

Human-aligned Cognitive Models

We released VSM-ACTR 2.0 (refer to VSM-ACTR below), which is a rule-based ACT-R cognitive decision-making model for manufacturing decision-making that implements multiple problem-solving strategies, through a combination of production rules. VSM-ACTR 2.0 has incorporated the meta-cognitive processes that reflect on and evaluate the progress of chosen strategies—with an emphasis on headcount cost evaluation, through a reward structure that enables a process akin to reinforcement learning. This system allows the model to dynamically assess the impact of headcount costs on decision-making outcomes, computing a reward or penalty for each decision cycle. These rewards or penalties then propagate back to the initial production rule that initiated the decision cycle, thereby dynamically adjusting the utility of each decision-making strategy.

VSM-ACTR 2.0 integrates the prototypical decision process with insights into how cognitive models represent different levels of expertise [14, 54], categorizing users into three levels of expertise: novices, intermediates, and experts. Novices engage in decision-making using intuitive deliberative chunks. Intermediates can manage key metrics such as CT and OEE but struggle with the systematic analysis of intertwined variables. Experts, on the other hand, make judgments systematically. The cognitive model employs three types of knowledge chunks: decisions, decision merits, and goals. The 'decision chunk' encodes eight slots including reduction time (goal), decision-making state (novice, intermediate, expert), and related variables. The 'decision merits chunk' holds information on sector weights, defect increases by sector, and comparative defect rate increases. The 'goal chunk' captures the initial production conditions and the ultimate goal of achieving the optimal decision. In addition, the model uses 18 procedural rules driven by goal-focused objectives across 20 states, covering actions such as choosing strategies, actions, working memory management, decisions, and evaluations.

Production Rule Sets

Three sets of production rules represent the decision-making behaviors of novice, intermediate, and expert decision-makers. These sets comprise a total of 18 rules, each driven by goal-focused objectives across 20 states.

We use the expert production rule set as an example (figure 3), once the decision-choice center decides to activate a set of expert decision productions, the process begins by perceiving the problem and retrieving related decision-making metrics from chunks. The imaginal buffer then acts as a working memory platform, holding and manipulating relevant information during the decision-making process. It allows the model to construct new mental representations or modify existing ones based on incoming data or problem-solving needs. This involves using the imaginal buffer to assess the relationships between the decision target and decision metrics, particularly considering the impact of each sector's weight on the defect rate change, and determining the final defect rate increase for each sector. These results are stored in the imaginal buffer and later retrieved for comparison. This enables the model to select the sector with the lowest defect increase. After one decision-making cycle, the model evaluates the headcount cost, rewarding or penalizing the entire process based on the evaluation results and decision strategy used before looping back to the next decision-making round.

Level of Expertise Mechanism

The model can learn while performing tasks through a mechanism leading to varying levels of expertise, as shown in Figure 4. The model mimics human decision-making behavior through differentiating knowledge representations. **Declarative Memories**: These memories store knowledge that aligns with human intuition and expertise gained from the VSM. For example, the green triangles in the figure represents a portion of the intuition used by novice decision-makers. **Production Rules**: These rules capture the rational decision-making processes observed in human subjects. The green lines illustrate how the imaginal buffer retrieves relevant portions of the novice declarative memory and feeds them to the novice production rule set. Intermediate and expert decision-making levels follow the

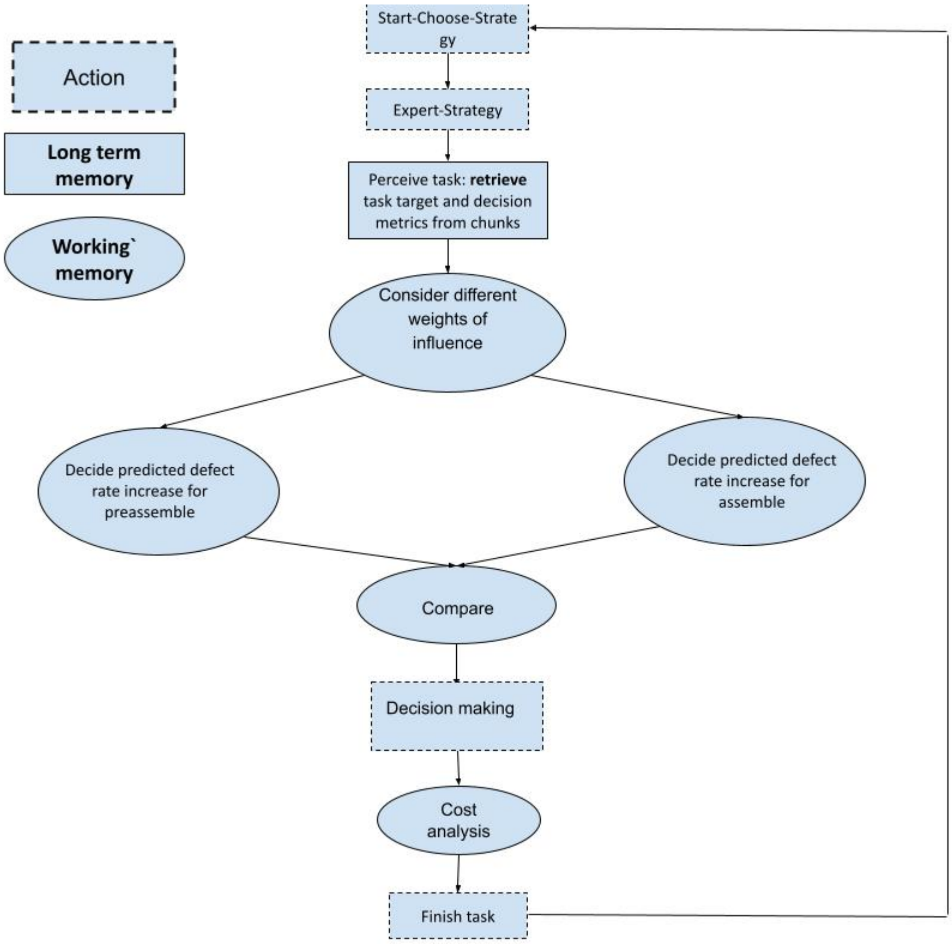

Fig. 3. Production rules control structure for expert decision making and their use of the ACT-R Goal and Imaginal buffers

same principle. Red and blue shapes represent their respective declarative memory chunks, and the corresponding colored arrows show the flow of information through their production rule sets. Finally, the goal buffer uses the "goal focus" command to manipulate the different phases of the task.

Beyond mimicking human behavior, the model also simulates the learning progress achieved by the **Decision-Choice Control**, which manages errors, learning, and memory through utility learning and reinforcement rewards. Novice decision-making starts with a utility base and includes a noise setting. The intermediate and expert production rules receive rewards when the corresponding decision-making results are achieved. The utility of these production rules updates is based on the rewards received and the retention of memory, which depends on the time passed since the rule last fired.

Reinforcement Mechanisms in Production Systems

Dopaminergic signals are believed to transmit reinforcement information to the corpus striatum [71], traditionally signaling reward-related activities. However, these signals are now understood to represent the error signal in the temporal difference (TD) algorithm from reinforcement learning [77], which is applied in ACT-R's learning mechanism. As expressed in Eqn. 1, Each production rule in the ACT-R model has a utility—a value or strength—associated with it, which is updated using the TD algorithm:

$$Eqn. 1: \; U_i(n) = U_i(n-1) + \alpha \left[R_i(n) + U_i(n-1)\right]$$

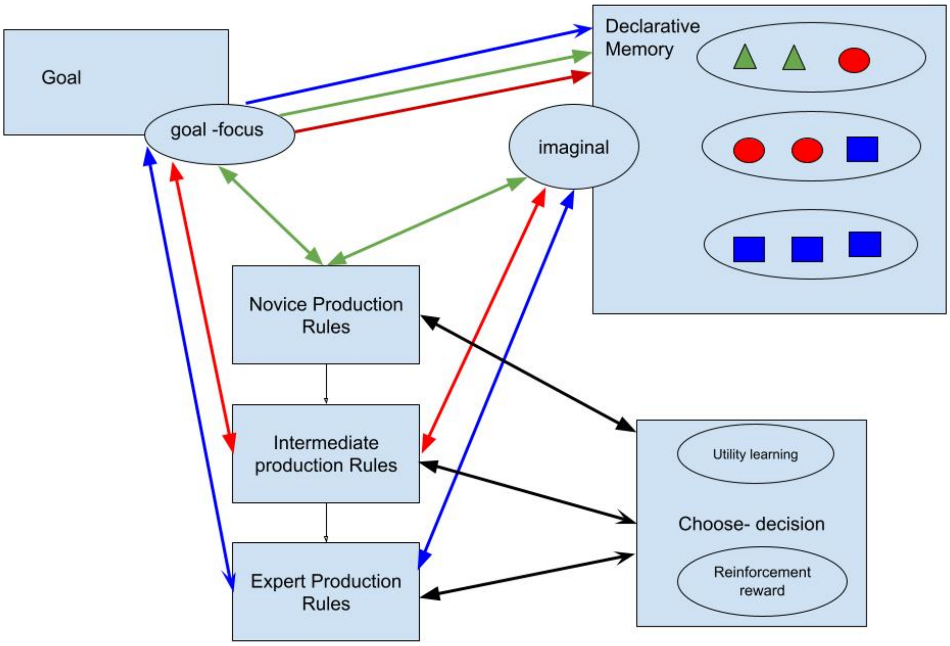

Fig. 4. Level of expertise mechanism in VSM-ACT-R

where $U_i(n)$ represents the value or utility of some item i (e.g., a production) after its n-th occurrence, and $R_i(n)$ represents the reward received on the n-th occurrence. The parameter α $(0 < \alpha < 1)$ controls the learning rate.

A key strength of the TD algorithm is its ability to propagate rewards back to earlier critical productions, through a chain of productions, influencing their utilities. This mechanism is tied to the widely-used 'softmax' function, which is also integral to ACT-R's production selection, as expressed in Equation 2. After propagation, if multiple productions compete with expected utility values U_j, the probability of of selecting production i is given by:

$$Eqn.\,2: \ \text{Probability}(i) = \frac{e^{U_i/\sqrt{2s}}}{\sum_j e^{U_j/\sqrt{2s}}},$$

where the summation over j is over all the productions that currently have their conditions satisfied; and s is the noise.

To understand the dynamics of the learning mechanism, consider a scenario involving penalties within a decision-making process shown in Figure 5, where p represents productions. The reward function $R(s, f(x))$ calculates the reward at the end of one decision-making round. This function takes two parameters: S, representing the strategy used, and $f(x)$, the outcome of the cost analysis, resulting in either a reward or a penalty. In one decision round, a penalty of -2 is computed due to the use of a novice strategy coupled with an inefficient cost. Factoring in the memory retention effect after a 0.5 time step, the subsequent penalty calculation modifies the impact of the decision:

$$R(S, x) - 0.5\,\text{time-steps} = -2.5$$

$$U(7) = U(6) + \alpha\,[R(S, x) - 0.5\,\text{time step} + U(6)] = -1.02$$

$U(7)$ represents the utility of novice strategy production at the seventh occurrence of firing. While $U(6)$ represents the utility at sixth occurrences; α is set at 0.2, based on the learning rule from [84]. This framework allows penalties to retroactively influence previous decisions, thus shaping the model's strategic choices in subsequent rounds.

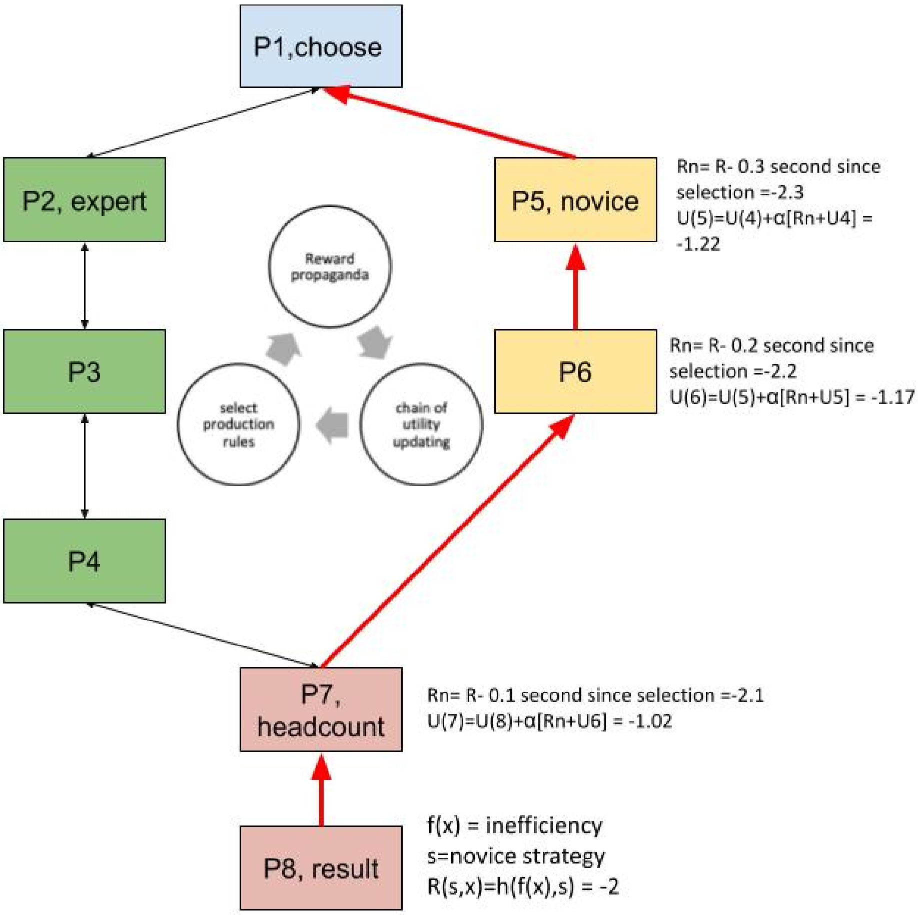

Fig. 5. reinforcement reward and utility propogate

VSM-ACTR model evaluation
We ran the VSM-ACTR model across 2012 decision-making trials and 32 problem sets to analyze its behavior [68]. Each model run comprised 15-16 trials until the model reached a more stable expert behavior. We encoded decision types as 0, 1, and 2 for novice, intermediate, and expert strategies, respectively.

To assess learning, individual differences, and progression, we initially used descriptive statistics to chart the average progression of decision types over 16 trials. We then employed a mixed linear model to evaluate the influence of trial numbers on decision types, incorporating repeated measures and random effects to account for individual variance. Additionally, an ordered logistic regression analyzed the relationship between the number of trials and the learning progression from novice to expert.

The results of the descriptive statistics demonstrate a significant positive impact of trial exposure on decision-making progression, evidenced by a coefficient of 0.086 ($P < 0.05$). A mixed linear model regression confirms the effect of trials on decision-making and further reveals a variance of 0.007 in the random group effects. This indicates that while there are differences between groups, these differences are relatively small, suggesting that the trials themselves predominantly explain the variability in decision type.

Threshold analysis using ordered logistic regression reveals significant transition thresholds. The transition from novice to intermediate has a significant threshold of 0.88 ($P < 0.05$), indicating a challenging progression to higher

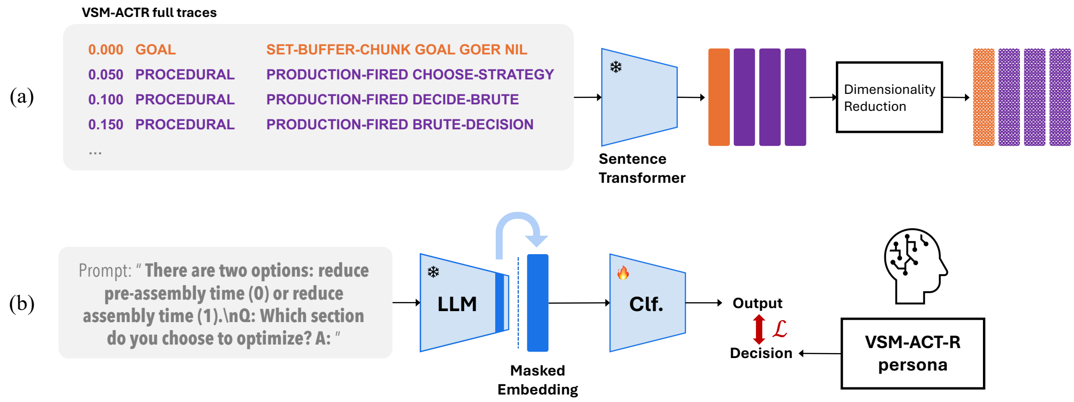

Fig. 6. (a) Obtaining decision representations from VSM-ACT-R. (b) LLM feature extraction for behavior prediction.

decision-making skills. In contrast, the transition from intermediate to expert shows a significantly lower threshold of 0.1 (P = 0.021), suggesting it is easier to progress from intermediate to expert than from novice to intermediate.

Comparing these results to the earlier VSM-ACTR 1.0 version [90], it's find that the transition from intermediate to expert has become significantly more pronounced. This change is attributed to enhancements in headcount cost evaluation factors that have reinforced the progression from intermediate to expert levels.

The `LLM−ACTR` *Framework*

Figure 1 illustrates the approach to creating `LLM−ACTR`, which begins with the collection of task data and documentation. The task procedures are then modeled using ACT-R, employing stochastic simulations to analyze these tasks on a large scale. After the simulation phase, the generated synthetic data is semi-automatically distilled and combined with prompt requests. This data is subsequently used to infuse into an open source LLM through fine-tuning, resulting in a type of cognitive LLM, named `LLM−ACTR`.

Selecting Salient Decision Information The decision-making process demonstrates cognitive reasoning through VSM-ACTR's traces, which capture the reasoning steps in real-time using a concurrent protocol (see Example ACT-R Decision Trace in the Appendix). These traces log the cognitive operations executed by various modules at each decision point, including the activation of the goal module to drive decisions, the use of the imaginal buffer for accessing working memory, procedural memory matching and firing, and utility updating driven by reinforcement learning, along with the decision actions.

Reserving information from ACT-R model's decision-making traces poses challenges. A single decision-making round can generate a vast number of lines of traces, each timestamped as frequently as every 5 milliseconds. Deciding which lines to select—or whether to preserve all lines—requires a balance between minimizing information loss and reducing computational costs. The rationale for choosing outputs from specific modules as reliable sources within the decision representation lies in their clear correspondence to deterministic cognitive processes. The rationale for preserving all traces involves processes of semantic embedding extraction and dimensional reduction.

The information used to augment decision-making in this study focuses on distilling macro-level cognitive processes related to executive function [28], capturing the evolution of decision-making results across trials and how decisions adapt through learning and experience. Furthermore, the decision actions are categorized into strategy levels (novice, intermediate, expert), reflecting the learning phases. Neurologically, as cognitive strategies evolve from novice to expert, there is a corresponding increase in the efficiency and effectiveness of neural circuits in the prefrontal cortex and basal ganglia in humans (see paragraphs: Implementing a reinforcement-learning mechanism in a production system framework).

Learning an Embedding Space of Decision Traces The next step involves converting the traces into tensors that the LLM can process. This study explores two approaches: one uses selected traces, and another uses full traces.

The selected traces are components distilled from macro-level cognitive processes related to executive function. This process requires human involvement to log decision results and strategy traces, which are then numerically encoded. For instance, '0' represents a decision for reduced time in preassembly section, and '1' for assembly. These data are subsequently fed into the neural network as single vectors.

In contrast, the holistic traces approach (see Figure 6a) retains both macro- and micro-level cognitive processes, with the latter including metacognition [59]. Metacognition involves an awareness and understanding of one's own cognitive processes, as exhibited through model traces that demonstrate the use of the imaginal buffer for accessing working memory, procedural memory matching and firing, headcount cost analysis, and the assessment of strategy effectiveness.

The investigation begins with the transformation of full traces from VSM-ACTR, representing both cognitive and metacognitive processes, into a format that balances information retention with computational efficiency. Cognitive reasoning traces for each task are processed through a sentence transformer to obtain semantic embeddings for each timestamp. A Sum of Ranked Explanatory Effects (SREE) analysis is then applied to determine the number (N) of principal components that account for at least 70% of the variance. Finally, these embeddings are reduced to N dimensions using Principal Components Analysis (PCA) [1].

Injecting Decision Information into LLMs With the VSM-ACTR model, which represents human-like cognitive reasoning in repeated decision-making tasks, this section outlines the experimental settings for fine-tuning of the `LLM-ACTR` framework. Fine-tuning, sometimes referred to as transfer learning, involves optimizing all model weights for the given task. As shown in figure 7, the process includes parsing consistent template prompts that reflect the decision making task into an open-source LLM, aligning the task for the cognitive model Using the LLM as the base model to access the last hidden layer and obtain masked embeddings, constructing a classification layer with softmax activation on top of the base model, using targets containing the salient decision representations of the cognitive model and features from the masked embeddings of the base LLM, and fine-tuning the LLM for classification using the LORA method. The key points are: (1) The targets decode the salient decision information from the cognitive model. (2) Use the final layer of contextualized embeddings in transformer-based LLMs, generated through the attention block mechanism. The attention block, a key feature of transformers, distinguishes them from other architectures like recurrent neural networks [31]. It creates embeddings that capture the in-context meaning of tokens by recombining them with other tokens' embeddings. Successive attention blocks further refine these embeddings, producing multiple layers of abstraction. The final layer, a blend of these refined embeddings, is used in this pipeline because it offers the richest semantic information while balancing minimal information loss and reduced computational costs for fine-tuning. (3) Use Low-Rank Adaptation (LoRa) for its efficiency in fine-tuning, reducing the computational resources and time required while maintaining high model performance [35].

Experiments

Problem Setting

As an instantiation of the problem definition, above, our manufacturing line has two sections with potential defect sources: pre-assembly and assembly. Pre-assembly takes 40 seconds with an OEE rate of 88%, while assembly takes 44 seconds with an OEE rate of 80.1%. To reduce total assembly time by 4, we must identify which section can be shortened with minimal defect increase. We note that reducing cycle time will also lead to an increase in headcount costs.

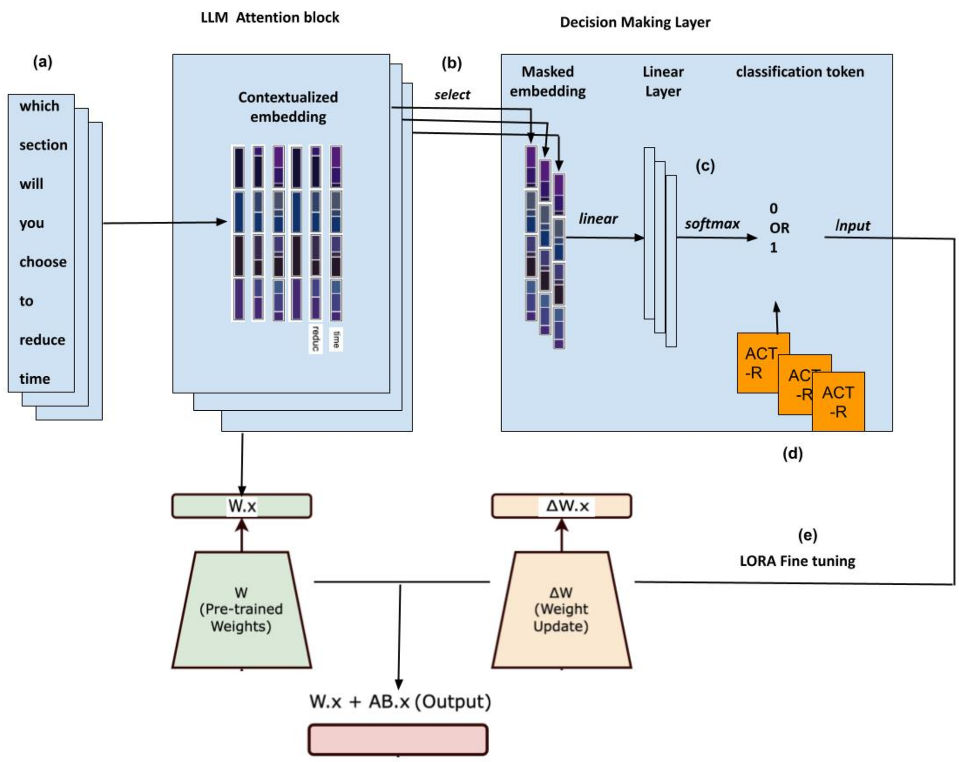

Fig. 7. Finetuning pipeline.

Implementation Details

The LlaMa-2 13B model was chosen as the foundation for this research because of its demonstrated effectiveness and efficiency in NLP tasks (Huang et al., 2024). As a state-of-the-art large language model, LlaMa has been trained on trillions of tokens from publicly available datasets. Unlike other transformer-based models such as the GPT family, which can only be accessed at the user's end, LlaMa's architecture, including its pre-trained weights, is fully accessible. Furthermore, its proven capability to extract the last hidden layer for predicting behavioral discrepancies has been provided (Binz and Schulz, 2024). These attributes collectively establish LlaMa-2 13B as an optimal choice for this study.

To determine the dataset size that can effectively perform the task while balancing efficacy and resource limitations, we referred to Kumar et al. [44], who showed evidence that LlaMa-2 13B achieves F1 scores above 0.9 in resource-limited text classification tasks, with datasets as 1,000 rows per class. Based on this, we developed the dataset size for fine-tuning as *N* (number of classes) * 1,000. The ACT-R dataset for binary decision-making classification contains 2,012 decision-making trials, Obtained by running the developed ACT-R model across 32 problem sets, each ACT-R persona was run for 15-16 trials until more stable expert behavior was achieved [68].

Baseline Models

This study compared the goodness-of-fit and prediction accuracy of the resulting models using holdout data against two baselines: a random guess model and LlaMa without fine-tuning, obtained by reading out log-probabilities of the pre-trained LlaMa.

A random guess model serves as the most basic form of chance level baseline and represents the simplest hypothesis for model comparison. In psychological interdisciplinary experiments, control conditions often employ random responses to distinguish the effects of treatment from chance [27]. This approach allows assessing the extent to which decisions are influenced by knowledge versus being purely stochastic.

On the other hand, using LlaMa without fine-tuning as a baseline provides a reference point to measure the impact of fine-tuning on the model's performance. This comparison reveals how much the model 'learns' from the fine-tuning process compared to its generic, pre-trained state.

Research Questions

Based on our framework's components, we identify a set of research questions that we answer through experiments.

1. *What are the properties of a useful neural network representation of the decision-making process in Cognitive Architectures?*

Answering this question sets the groundwork for developing a context-aware domain knowledge base for augmenting decision-making in LLMs.

2. *What level of complexity in behavior representation can LLMs effectively capture?*

Previous research has used LLM conceptual embeddings to predict human behavior based on past behavioral studies [13], confirming LLMs' ability to replicate known human patterns. However, high costs and extensive data collection efforts limit this method. By incorporating cognitive model simulations, the study seeks to address these limitations and broaden the investigation to determine the extent to which LLMs can reproduce decision-making knowledge. This will, in turn, help define the depth of decision-making domain knowledge that can be effectively integrated with the innate learning capabilities of LLMs.

3. *Can we inform the LLM with knowledge about the reasoning process of the cognitive architecture?*

Inspired by previous work on knowledge-injection [52, 62], answering this question offers insights into knowledge transfer from domain-specific bases to LLMs and evaluates its impact on performance in holdout tasks. The method for addressing RQ1 was introduced in the first two sections of our approach framework.

Feature Extraction for Behavior Prediction

To answer RQ2: What level of complexity in behavior representation can LLMs effectively capture? Building on previous research that used conceptual embeddings from LLMs to predict human behavior with historical behavioral data [13], we adopted the same method of LLM feature extraction for behavior prediction [38]. We created datasets consisting of last contextual embeddings as features and the corresponding different levels of VSM-ACTR decision actions representations as targets. We obtained embeddings by passing prompts that included all the information that VSM-ACTR had access to on a given trial through LlaMa and then extracting the hidden activations of the final layer, as shown in Figure 6b.

The first dataset used features extracted from prompts (see Appendix: LLMs system prompt templates) identical to the VSM-ACTR task, with targets being the VSM-ACTR decision-making results, where '0' indicates reduced time in preassembly and '1' indicates assembly. The second dataset's prompt template added an explanation of the strategy adopted by VSM-ACTR and used compound targets comprising both the decision-making results and the strategies reflecting the learning trajectory (novice, intermediate, and expert). The targets were encoded as follows: 0, 1, and 2 for preassembly choices using novice, intermediate, and expert strategies, respectively, and 3, 4, and 5 for assembly choices following the same pattern. With these two datasets, we fitted a regularized logistic regression model using 10-fold cross-validation for dataset 1 and multinomial regression using 10-fold cross-validation with L2 regularization for dataset 2. Model performance was assessed by measuring the goodness of fit through negative log-likelihood (NLL) and the predictive accuracy of hold-out data.

Fine Tuning for Knowledge Transfer

To answer RQ3: whether LLMs can be informed with knowledge about the reasoning processes of cognitive architecture—we use the fine-tuning approach of `LLM-ACTR` Framework. The fine-tuning process employs Cross-Entropy as the loss function and uses Adam optimization. Training involves a train test split of 0.2 and uses a batch size of 5 for both training and validation phases. The learning rate is set to 1e-5, with the training spanning across 10 epochs. To ensure regularization and prevent overfitting, a weight decay of 0.01, and a dropout of 0.5 are applied, and gradient accumulation is set to 2. Last but not least, gradient clipping is employed to maintain a maximum gradient norm of 1.0 for gradient explosion control. We evaluate the model fitting and generalization quality using training loss and validation loss across epochs, then compare the goodness of fit and prediction accuracy of the hold-out data against the baseline models.

Results

We present the results of addressing the research questions and subsequently report preliminary experimental findings on injecting holistic traces of VSM-ACTR into `LLM-ACTR`.

Finding Useful Decision Process Embeddings

The approach of distilling macro-level cognitive processes related to executive function captures the evolution of decision-making results across trials and how decisions adapt through learning and experience, all represented as a sequential single vector. This format facilitates ease of use for downstream tasks involving knowledge transfer. However, this method retains only partial cognitive decision-making knowledge.

In contrast, the holistic semantic preservation approach encompasses both macro and micro-level cognition processes. However, the embeddings produced vary in shape due to the individual differences in traces originating from stochastic simulations. They cannot be directly fed into neural networks for downstream tasks. Nevertheless, the first two principal components of the reduced embeddings, which correspond to the semantic mapping of ACT-R's components—including procedural, imaginal, goal knowledge, utility updating, and decision-making—are detailed in Figure 8.

The MANOVA analysis was conducted to assess the overall effect of the independent variables, which include label categories or ACT-R components, on the combined dependent variables—components of reduced embeddings. This analysis reveals a significant relationship with the semantic mapping of ACT-R's components. For instance, the extremely low Wilks' lambda value (0.0004) suggests that the label or ACT-R component categories explain nearly all the variance in the dependent variables, indicative of a strong group effect. The statistical tests applied—Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root—all demonstrate strong significance, as evidenced by the extremely low p-values across all tests. These findings highlight that the principal components retained in the PCA successfully capture the essential variance related to these cognitive processes. This result validates that ACT-R reasoning process can be mapped through neural network.

Assessing Behavior Complexity Captured by LLMs

Table 1 shows that `LLM-ACTR` captures a single facet of decision-making, achieving an average accuracy of 0.64 across 10 validation folds in the holdout task. When decision-making targets involve multiple facets—encompassing both choices and strategies that shape the learning trajectory—the accuracy decreases to 0.42. While this reduction suggests that capturing complex decision-making processes is less accurate, the results still show promise in handling these complexities. However, the Negative Log-Likelihood (NLL) reveals greater predictive uncertainty for multifaceted decision-making processes, as evidenced by a significantly higher NLL of 1.18 compared to 0.65 in single-facet scenarios.

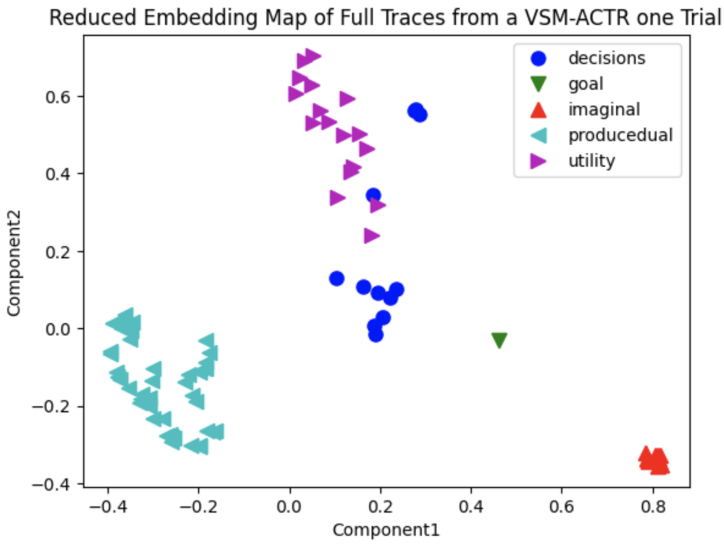

Fig. 8. ACTR embedding mapping

Table 1
Evaluation for Single and Multi Facets Targets

Target Type	**NLL**	**Accuracy**
Single Facet Target	0.63	0.64
Multi Facets Target	1.18	0.42

Table 2
Comparison of VSM-ACTR with baselines

Model	**NLL**	**Accuracy**
Chance-level	0.6931	0.4826
LlaMa	1.1330	0.3564
`LLM-ACTR`(ours)	**0.6534**	**0.6576**

Injecting LLMs with CA Decision Process

We first report training and validation losses, across 10 epochs, to reveal the fine-tuned model's learning and generalization behavior. Initially, the training loss begins at approximately 0.73, with a slight fluctuation observed in subsequent epochs, peaking around epoch 2 and showing a notable dip at epoch 7. In contrast, the validation loss starts at around 0.64 and remains remarkably stable throughout the epochs. This consistency in validation loss, coupled with a generally downward trend in training loss after its initial variations, suggests that the model is learning effectively. The overall trend indicates an improvement in model performance over time, reflecting its capability to generalize well on unseen data.

We then report the comparison of the `LLM-ACTR` with the baseline models on goodness of fit using negative log likelihood (NLL) and accuracy score for hold-out data. The `LLM-ACTR` model demonstrates significantly better performance across all metrics compared to the LlaMa-only model, highlighting its effectiveness in decision-making tasks involving sequential cognitive reasoning. Additionally, the LlaMa-only model performs worse than the chance-level model. This underscores the necessity of fine-tuning pre-trained language models like LlaMa to adapt them to specific human-aligned repeated decision-making tasks.

Preliminary Experimental Results on Injecting Holistic Traces of VSM-ACTR into `LLM-ACTR`

Followed results that validate VSM-ACTR reasoning process can be mapped through the neural network, with this hypothesis, we conducted two preliminary experiments on injecting holistic traces of VSM-ACTR into LLM in this decision-making task.

In the first preliminary experiment, we used the method of feature extraction, and addressed the issue of ragged tensors by employing padding with value imputation (Figure 9). We then integrated the 240 full cognitive reasoning traces from the VSM-ACTR model with LLM using embedding concatenation and conducted feature extraction for behavior prediction. Specifically, we transposed the reduced embeddings from each cognitive model run into a (1, X) dimension tensor and subsequently concatenated this with the LLM's last contextual embedding from the same prompt. These concatenated embeddings served as resources for predicting decision-making within the VSM-ACTR model. The prediction targets were multifaceted, including both the decision-making results and the strategies used. We employed a train-test split of 0.4 and conducted multinomial regression with L2 regularization, using two-fold cross-validation to compare the prediction accuracy and goodness of fit, as measured by negative log-likelihood, between concatenated embeddings and LlaMa embeddings alone. The results indicated that the prediction accuracy for both datasets was 0.71, with the concatenated embeddings showing a slightly better negative log-likelihood of 0.9535 compared to 0.9553 for LlaMa-only embeddings.

The results suggest no significant improvement in behavior prediction when combining masked embeddings versus using LlaMa embeddings alone. One possible explanation is the relative scale of the VSM-ACTR reduced embeddings compared to those of LlaMa, which is disproportionately small (1:10). Consequently, the LlaMa embeddings may dominate the decision-making process within the model due to their larger scale. A potential solution could be to generate more VSM-ACTR model traces with more variation, thereby enhancing the scale and variability of its features. Also, the method we use to handle ragged tensors—padding followed by value imputation—could potentially dilute the VSM-ACTR embeddings and reduce their accuracy. Finding an alternative method to preserve the full embeddings from VSM-ACTR may potentially improve the results. Lastly, the limited dataset size could be influencing the results. The preliminary test used only 240 complete traces. Expanding the dataset may provide more insights into the performance of the proposed approach.

The second preliminary experiment employs `LLM-ACTR` with a modification that incorporates the vector representation of `LLM-ACTR`'s full decision-making traces into the hidden state of LLMs during fine-tuning (Figure 10). These vectors are obtained using our holistic semantic preservation approach, which begins with processing 240 ACT-R traces through a sentence transformer. Next, Principal Component Analysis is applied for semantic abstraction at each timestamp of the trace, followed by tensor concatenation. We addressed the ragged tensor by padding, then calculated the standardized mean values of these tensors and integrated the normalized vectors into one of the hidden layers of the LlaMa 7B model, using a scaling factor. We switched to a smaller size of LlaMa to strike a balance between the computational costs of backpropagation when modifying the model's hidden layers and the overall efficacy of the base model.

Subsequently, the LlaMa model with the modified hidden layer is fine-tuned with 2012 data points for the binary classification task. The vectors from VSM-ACTR are set to be non-trainable to ensure the preservation of their integrity and to prevent gradient explosion. The results show that prediction accuracy remained unchanged. However, the negative log-likelihood improved, as illustrated in Figure 11. The addition of the vector representation of VSM-ACTR's holistic traces during fine-tuning resulted in a decreased mean Negative log-likelihood value and reduced NLL variance across 10 epochs, demonstrating better model fitting and stability compared to fine-tuning only.

The improved model fitting implies that the LLM becomes more effective in capturing the underlying patterns in the data. Furthermore, stability in performance enhances the trustworthiness of the model. However, despite these improvements in model stability and fitting, the prediction accuracy remained unchanged. This lack of improvement indicates that further research is needed to understand the factors limiting accuracy and to explore additional modifications, such as vector optimization, that could translate enhancements in stability and fitting into tangible gains in prediction performance.

Discussion and Conclusion

Main Insights/Takeaways Resolving the dichotomy between the human-like yet constrained reasoning processes of CAs and the broad, often noisy inference behavior of LLMs remains a challenging but exciting pursuit. This is crucial for enabling reliable machine reasoning capabilities in production systems. This study introduces , a novel neuro-symbolic architecture designed to enhance human-aligned and versatile decision-making by integrating the

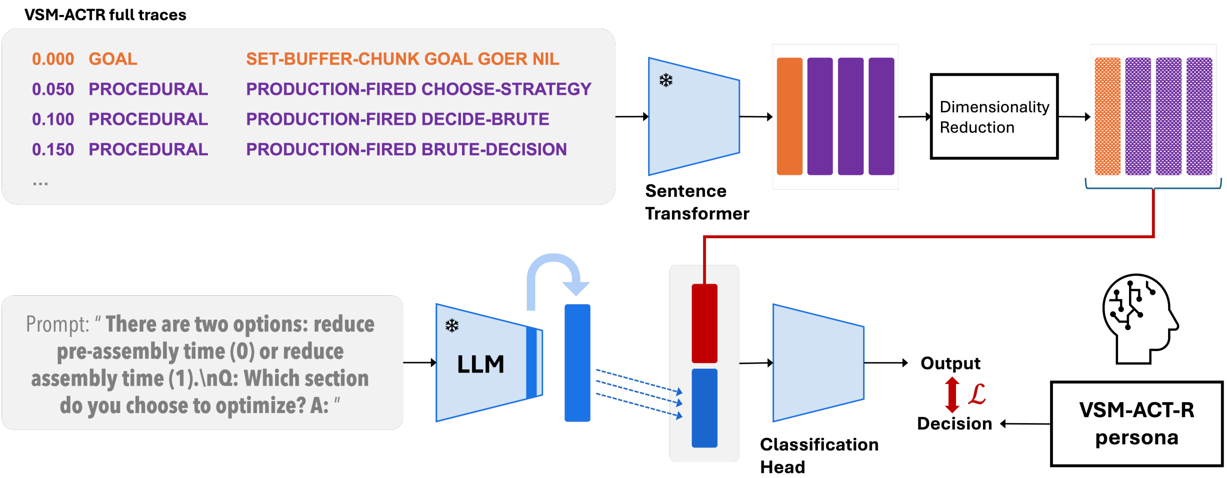

Fig. 9. Infusing holistic VSM-ACTR traces through semantic abstraction, feature extraction,and concatenation

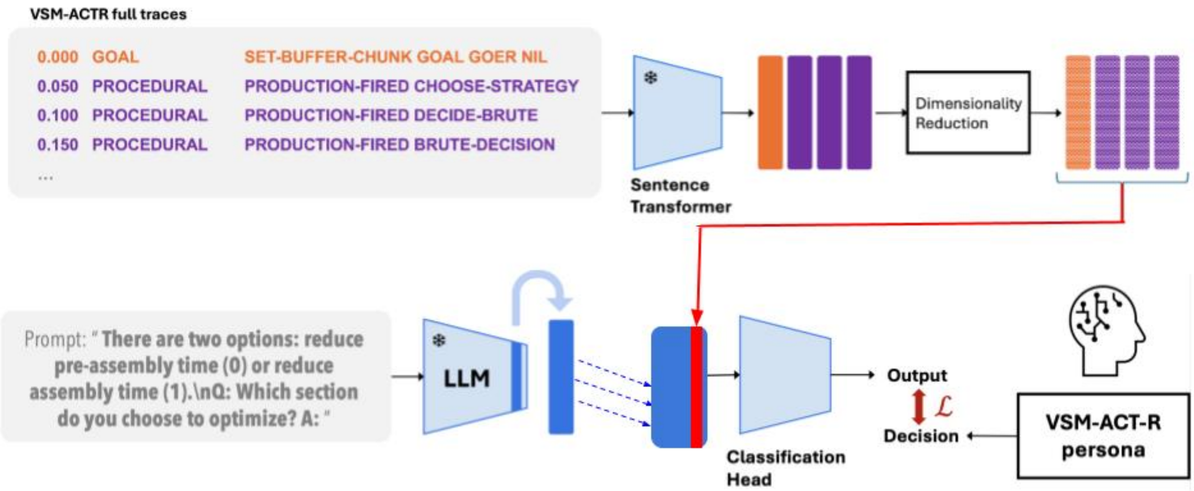

Fig. 10. Infusing holistic VSM-ACTR traces through fine-tuning with vectors from holistic VSM-ACTR traces

ACT-R model's cognitive process with LLMs. Our framework extracts and embeds an ACT-R model's internal decision-making processes as latent neural representations based on using traces of its performance, then injects this information into trainable LLM adapter layers, and finally fine-tunes the LLMs for downstream prediction tasks. LLM-ACTR addresses the data scarcity issue often encountered in research aimed at aligning LLMs with human reasoning. Our approach demonstrates improved grounded decision-making capabilities compared to LLM-only baselines that leverage chain-of-thought reasoning strategies.

We explore distilling latent representations. The findings show that distilling macro-level cognitive processes preserves high-level neural symbolic knowledge, aiding downstream tasks but only partially capturing decision-making knowledge. A holistic semantic preservation approach, covering both cognitive and metacognitive processes, better retains full neural symbolic semantics with low computational costs. However, challenges with ragged tensors in downstream tasks require further research. We then use a VSM-ACTR cognitive model, developed for a manufacturing design task, to distill its macro-level cognitive processes as domain knowledge. This knowledge was then employed in both a feature extraction for behavior prediction method and a fine-tuning pipeline to investigate the

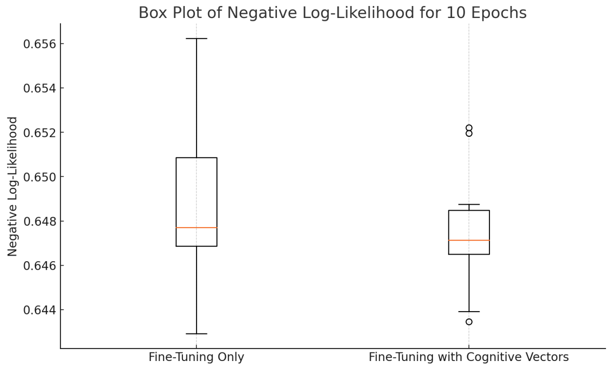

Fig. 11. Comparison of NLL across 10 epochs for fine-tuning only and fine-tuning with vectors from holistic VSM-ACTR traces

LLM's capabilities in (1) capturing the complexity of behavioral representations and (2) determining whether and how the LLM can be informed by the reasoning processes inherent in the cognitive architecture.

The results show that (1) prompt embeddings generated through feature extraction capture repeated human-aligned decision-making and the overall structure of learning. However, they struggle to capture complex decision-making rationales, such as the strategies employed. (2) Fine-tuning effectively informs the LLM with knowledge of decision-making tasks involving sequential cognitive reasoning.

Limitations The limitation lies in achieving close human alignment with the VSM-ACTR model. While the model is currently tuned to reflect general human learning behavior patterns in decision-making, it still requires more granular human data for cognitive fine-tuning. The closer the VSM-ACTR model aligns with human behavior, the more accurately it can represent human decision-making processes.

Future Work To address this limitation, the project aims to enhance the human alignment of the cognitive model by fine-tuning it with granular human data. This will be followed by integrating the full semantic space of ACT-R into the LLM's decision-making processes as explored in the preliminary experiments. The strategy of using semantic abstraction from cognitive model trace lines demonstrates the potential. However, the preliminary experimental results highlight challenges in integrating such knowledge, which are worth further exploration. Grounding LLMs in a robust cognitive framework while minimizing computational costs could pave the way for effectively scaling the infusion of superior domain knowledge into LLMs.

Credit Author Statement

Siyu Wu: Conceptualization, Methodology, Software, Experiments, Writing - Original Draft, Writing - Review & Editing. Alessandro Oltramari: Conceptualization, Funding Acquisition, Methodology, Writing - Review & Editing. Jonathan Francis: Methodology, Experiments, Writing- Review & Editing. C. Lee Giles: Conceptualization, Writing - Review & Editing. Frank E. Ritter: Writing - Review & Editing.

References

[1] H. Abdi and L.J. Williams, Principal Component Analysis, *Wiley Interdisciplinary Reviews: Computational Statistics* **2**(4) (2010), 433–459.
[2] S. Abdurahman, H. Vu, W. Zou, L. Ungar and S. Bhatia, A Deep Learning Approach to Personality Assessment: Generalizing Across Items and Expanding the Reach of Survey-Based Research, *Journal of Personality and Social Psychology* (2023), Advance online publication. doi:10.1037/pspp0000480.
[3] A. Aka and S. Bhatia, Machine learning models for predicting, understanding, and influencing health perception, *Journal of the Association for Consumer Research* **7**(2) (2022), 142–153. doi:10.1086/718456.
[4] J.R. Anderson, *The atomic components of thought*, Erlbaum, Mahwah, NJ, 1998, pp. 255–296.
[5] J.R. Anderson, *How can the human mind occur in the physical universe?*, Oxford University Press, 2009.

[6] J.R. Anderson, S. Betts, D. Bothell, R. Hope and C. Lebiere, Learning rapid and precise skills, *Psychological Review* **126** (2019), 727–760.

[7] J. Andreas, Language models as agent models, in: *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022, pp. 5769–5779.

[8] Authors, VSM-ACTR: Toward Using Cognitive Architecture for Manufacturing Solutions, in: *Accepted to the 17th International Conference on Social Computing, Behavioral-Cultural Modeling & Prediction and Behavior Representation in Modeling and Simulation (SBP-BRIMs)*, 2024.

[9] G. Bao, H. Zhang, L. Yang, C. Wang and Y. Zhang, LLMs with chain-of-thought are non-causal reasoners, 2024.

[10] B.J. Best and C. Lebiere, Teamwork, Communication, and Planning in ACT-R Agents Engaging in Urban Combat in Virtual Environments, in: *Proceedings of the 2003 IJCAI Workshop on Cognitive Modeling of Agents and Multi-Agent Interactions*, 2003, pp. 64–72.

[11] S. Bhattamishra, A. Patel, P. Blunsom and V. Kanade, Understanding in-context learning in transformers and LLMs by learning to learn discrete functions, *arXiv preprint arXiv:2310.03016* (2023).

[12] M. Binz and E. Schulz, Using cognitive psychology to understand GPT-3, *Proceedings of the National Academy of Sciences* **120**(6) (2023), e2218523120.

[13] M. Binz and E. Schulz, Turning Large Language Models into Cognitive Models, in: *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024, In press.

[14] S.B. Blessing and J.R. Anderson, How People Learn to Skip Steps, *Journal of Experimental Psychology: Learning, Memory, and Cognition* **22**(3) (1996), 576.

[15] D. Bothell, *ACT-R 7 Reference Manual*, 2017, Available at http://act-r.psy.cmu.edu/wordpress/wp-content/themes/ACT-R/actr7/reference-manual.pdf.

[16] T.B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal et al., Language models are few-shot learners, in: *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33, 2020, pp. 1877–1901.

[17] N. Chakraborty, M. Ornik and K. Driggs-Campbell, Hallucination Detection in Foundation Models for Decision-Making: A Flexible Definition and Review of the State of the Art, *arXiv* **2403.16527** (2024), 31. doi:10.48550/arXiv.2403.16527.

[18] H. Chen, S. Liu, L. Pang, X. Wanyan and Y. Fang, Developing an Improved ACT-R Model for Pilot Situation Awareness Measurement, *IEEE Access* **9** (2021), 122113–122124.

[19] J. Coda-Forno, M. Binz, J.X. Wang and E. Schulz, CogBench: A Large Language Model Walks into a Psychology Lab, in: *Accepted to the International Conference on Learning Representations (ICLR)*, 2024.

[20] J. Coda-Forno, M. Binz, Z. Akata, M. Botvinick, J. Wang and E. Schulz, Meta-in-context learning in large language models, in: *Advances in Neural Information Processing Systems*, Vol. 36, 2024.

[21] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu and Z. Sui, A survey on in-context learning, *arXiv preprint arXiv:2301.00234* (2022).

[22] M. Dorobantu, Human-level, but non-humanlike: Artificial intelligence and a multi-level relational interpretation of the imago Dei, *Philosophy, Theology and the Sciences* **8**(1) (2021), 81–107.

[23] V.G. dos Santos, G.L. Santos, T. Lynn and B. Benatallah, Identifying Citizen-Related Issues from Social Media Using LLM-Based Data Augmentation, in: *International Conference on Advanced Information Systems Engineering*, Springer Nature Switzerland, Cham, 2024, pp. 531–546.

[24] T. Eloundou, S. Manning, P. Mishkin and D. Rock, GPTs are GPTs: An early look at the labor market impact potential of large language models, *arXiv preprint arXiv:2303.10130* (2023).

[25] H. Fei, Y. Yao, Z. Zhang, F. Liu, A. Zhang and T.S. Chua, From Multimodal LLM to Human-level AI: Modality, Instruction, Reasoning, Efficiency and Beyond, in: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024): Tutorial Summaries*, 2024, pp. 1–8.

[26] J. Francis, N. Kitamura, F. Labelle, X. Lu, I. Navarro and J. Oh, Core challenges in embodied vision-language planning, *Journal of Artificial Intelligence Research* **74** (2022), 459–515.

[27] J. Gaab, J. Kossowsky, U. Ehlert and C. Locher, Effects and Components of Placebos with a Psychological Treatment Rationale—Three Randomized-Controlled Studies, *Scientific Reports* **9**(1) (2019), 1421.

[28] S.J. Gilbert and P.W. Burgess, Executive Function, *Current Biology* **18**(3) (2008), R110–R114.

[29] K.A. Gluck and R.W. Pew, *Modeling human behavior with integrated cognitive architectures: Comparison, evaluation, and validation*, Psychology Press, 2006.

[30] C. Gonzalez, J.F. Lerch and C. Lebiere, Instance-based learning in dynamic decision making, *Cognitive Science* **27** (2003), 591–635. doi:10.1207/s15516709cog2704$_2$.

[31] A. Graves, Sequence transduction with recurrent neural networks, *arXiv preprint arXiv:1211.3711* (2012).

[32] T. Hagendorff, S. Fabi and M. Kosinski, Human-like intuitive behavior and reasoning biases emerged in large language models but disappeared in ChatGPT, *Nature Computational Science* **3**(10) (2023), 833–838.

[33] H. Hanke and D. Knees, A phase-field damage model based on evolving microstructure, *Asymptotic Analysis* **101** (2017), 149–180.

[34] E. Hozdić, Smart Factory for Industry 4.0: A review, *International Journal of Modern Manufacturing Technologies* **7**(1) (2015), 28–35.

[35] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang and W. Chen, LoRA: Low-Rank Adaptation of Large Language Models, in: *International Conference on Learning Representations*, 2022. https://openreview.net/forum?id=nZeVKeeFYf9.

[36] Y. Hu, Q. Xie, V. Jain, J. Francis, J. Patrikar, N. Keetha, S. Kim, Y. Xie, T. Zhang, Z. Zhao et al., Toward general-purpose robots via foundation models: A survey and meta-analysis, *arXiv preprint arXiv:2312.08782* (2023).

[37] D. Huang, Z. Hu and Z. Wang, Performance Analysis of Llama 2 Among Other LLMs, in: *2024 IEEE Conference on Artificial Intelligence (CAI)*, 2024, pp. 555–559.

[38] Z. Hussain, M. Binz, R. Mata and D.U. Wulff, A tutorial on open-source large language models for behavioral science, *PsyArXiv preprint* (2023).

[39] H. Joshi and V. Ustun, Augmenting Cognitive Architectures with Large Language Models, in: *Proceedings of the AAAI Symposium Series*, Vol. 2, 2024. doi:10.1609/aaaiss.v2i1.27689.

[40] D. Kahneman, *Thinking, fast and slow*, macmillan, 2011.

[41] J.R. Kirk, W. Robert, P. Lindes and J.E. Laird, Improving Knowledge Extraction from LLMs for Robotic Task Learning through Agent Analysis, 2023.

[42] J.R. Kirk, R.E. Wray, P. Lindes and J.E. Laird, Improving Knowledge Extraction from LLMs for Task Learning through Agent Analysis, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38, 2024, pp. 18390–18398.

[43] T. Kojima, S.S. Gu, M. Reid, Y. Matsuo and Y. Iwasawa, Large Language Models are zero-shot reasoners, 2022, https://api.semanticscholar.org/CorpusID:249017743.

[44] A. Kumar, R. Sharma and P. Bedi, Towards Optimal NLP Solutions: Analyzing GPT and LLaMA-2 Models Across Model Scale, Dataset Size, and Task Diversity, *Engineering, Technology & Applied Science Research* **14**(3) (2024), 14219–14224.

[45] J.E. Laird, *The Soar cognitive architecture*, MIT Press, 2012.

[46] J.E. Laird, C. Lebiere and P.S. Rosenbloom, A Standard Model of the Mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics, *AI Magazine* **38**(4) (2017), 13–26.

[47] E. Lefever, A hybrid approach to domain-independent taxonomy learning, *Applied Ontology* **11**(3) (2016), 255–278.

[48] M. Leyer and S. Schneider, Decision augmentation and automation with artificial intelligence: Threat or opportunity for managers?, *Business Horizons* **64**(5) (2021), 711–724.

[49] H. Liang, X. Sun, Y. Sun and Y. Gao, Text feature extraction based on deep learning: a review, *EURASIP Journal on Wireless Communications and Networking* **2017** (2017), 1–12.

[50] R. Liu, J. Geng, J.C. Peterson, I. Sucholutsky and T.L. Griffiths, Large Language Models Assume People are More Rational Than We Really are, *arXiv* **cs.CL** (2024). https://doi.org/10.48550/arXiv.2406.17055.

[51] M.C. Lovett and J.R. Anderson, History of success and current context in problem solving: Combined influences on operator selection, *Cognitive Psychology* **31** (1996), 168–217.

[52] K. Ma, J. Francis, Q. Lu, E. Nyberg and A. Oltramari, Towards generalizable neuro-symbolic systems for commonsense question answering, *arXiv preprint arXiv:1910.14087* (2019).

[53] T. Manos, Value Stream Mapping-An introduction, *Quality Progress* **39**(6) (2006), 64.

[54] M.K. Martin, C. Gonzalez and C. Lebiere, Learning to Make Decisions in Dynamic Environments: ACT-R Plays the Beer Game, in: *Proceedings of the Sixth International Conference on Cognitive Modeling*, Carnegie Mellon University/University of Pittsburgh, Pittsburgh, PA, 2004.

[55] J. McCarthy, From Here to Human-Level AI, *Artificial Intelligence* **171**(18) (2007), 1174–1182.

[56] P.S. Meltzer, A. Kallioniemi and J.M. Trent, Chromosome alterations in human solid tumors, in: *The Genetic Basis of Human Cancer*, B. Vogelstein and K.W. Kinzler, eds, McGraw-Hill, New York, 2002, pp. 93–113.

[57] S. Moore, R. Tong, A. Singh, Z. Liu, X. Hu, Y. Lu and J. ... Stamper, Empowering Education with LLMs-The Next-Gen Interface and Content Generation, in: *International Conference on Artificial Intelligence in Education*, Springer Nature Switzerland, Cham, 2023, pp. 32–37.

[58] P.R. Murray, K.S. Rosenthal, G.S. Kobayashi and M.A. Pfaller, *Medical Microbiology*, 4th edn, Mosby, St. Louis, 2002.

[59] T.O. Nelson and L. Narens, Why Investigate Metacognition?, *Journal Name Here* **Volume Here**(Issue Number Here) (1994), Page Numbers Here.

[60] A. Newell, *Unified Theories of Cognition*, Harvard University Press, 1994.

[61] A. Oltramari, Enabling High-Level Machine Reasoning with Cognitive Neuro-Symbolic Systems, in: *Proceedings of the AAAI Symposium Series*, Vol. 2, 2023, pp. 360–368.

[62] A. Oltramari, J. Francis, F. Ilievski, K. Ma and R. Mirzaee, Generalizable neuro-symbolic systems for commonsense question answering, in: *Neuro-Symbolic Artificial Intelligence: The State of the Art*, IOS Press, 2021, pp. 294–310.

[63] OpenAI, About OpenAI, 2023.

[64] S. Papert, You Can't Think About Thinking Without Thinking About Thinking About Something, *Contemporary Issues in Technology and Teacher Education* **5**(3/4) (2005), 366–367.

[65] S. Prezenski, A. Brechmann, S. Wolff and N. Russwinkel, A cognitive modeling approach to strategy formation in dynamic decision making, *Frontiers in Psychology* **8** (2017), 1335.

[66] A.R. Rahani and M. Al-Ashraf, Production Flow Analysis Through Value Stream Mapping: A Lean Manufacturing Process Case Study, *Procedia Engineering* **41** (2012), 1727–1734.

[67] F.E. Ritter, F. Tehranchi and J.D. Oury, ACT-R: A cognitive architecture for modeling cognition, *Wiley Interdisciplinary Reviews: Cognitive Science* **10** (2023), 833–838.

[68] F.E. Ritter, M.J. Schoelles, K.S. Quigley and L.C. Klein, *Determining the number of model runs: Treating cognitive models as theories by not sampling their behavior*, in: *Human-in-the-loop simulations: Methods and practice*, Springer-Verlag, London, 2011, pp. 97–116.

[69] I. Roll, R.S. Baker, V. Aleven and K.R. Koedinger, A Metacognitive ACT-R Model of Students' Learning Strategies in Intelligent Tutoring Systems, in: *Intelligent Tutoring Systems: 7th International Conference, ITS 2004, Maceió, Alagoas, Brazil, August 30-September 3, 2004. Proceedings*, Springer Berlin Heidelberg, 2004, pp. 854–856.

[70] D.A. Rossit, F. Tohmé and M. Frutos, Industry 4.0: Smart Scheduling, *International Journal of Production Research* **57**(12) (2019), 3802–3813.

[71] T. Schönberg, N.D. Daw, D. Joel and J.P. O'Doherty, Reinforcement learning signals in the human striatum distinguish learners from nonlearners during reward-based decision making, *Journal of Neuroscience* **27**(47) (2007), 12860–12867.
[72] C.D. Schunn, M.C. Lovett and L.M. Reder, Awareness and Working Memory in Strategy Adaptivity, *Memory Cognition* **29** (2001), 254–266.
[73] S.A. Sloman, The empirical case for two systems of reasoning, *Psychological Bulletin* **119**(1) (1996), 3–22.
[74] A. Stocco, C. Sibert, Z. Steine-Hanson, N. Koh, J.E. Laird, C.J. Lebiere and P. Rosenbloom, Analysis of the human connectome data supports the notion of a "Common Model of Cognition" for human and human-like intelligence across domains, *NeuroImage* **235** (2021), 118035.
[75] T.R. Sumers, S. Yao, K. Narasimhan and T.L. Griffiths, Cognitive architectures for language agents, *arXiv preprint arXiv:2309.02427* (2023).
[76] R.S. Sutton and A.G. Barto, Toward a Modern Theory of Adaptive Networks: Expectation and Prediction, *Psychological Review* **88**(2) (1981), 135.
[77] R.S. Sutton and A.G. Barto, Reinforcement Learning, *Journal of Cognitive Neuroscience* **11**(1) (1999), 126–134.
[78] G. Tatiya, J. Francis, H.-H. Wu, Y. Bisk and J. Sinapov, Mosaic: Learning unified multi-sensory object property representations for robot perception, *arXiv preprint arXiv:2309.08508* (2023).
[79] A.J. Thirunavukarasu, D.S.J. Ting, K. Elangovan, L. Gutierrez, T.F. Tan and D.S.W. Ting, Large Language Models in Medicine, *Nature Medicine* **29**(8) (2023), 1930–1940.
[80] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave and G. Lample, LLaMA: Open and Efficient Foundation Language Models, 2023. https://api.semanticscholar.org/CorpusID:257219404.
[81] A. Tversky and D. Kahneman, Judgment under Uncertainty: Heuristics and Biases, *Science* **185** (1974), 1124–1131.
[82] K. Ulrich, D. Sartorius, S. Pearson and M. Jakiela, Including the Value of Time in Design-for-Manufacturing Decision Making, *Management Science* **39**(4) (1993), 429–447.
[83] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, Attention is all you need, in: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 6000–6010–. ISBN 9781510860964.
[84] A.R. Wagner and R.A. Rescorla, Inhibition in Pavlovian conditioning: Application of a theory, in: *Inhibition and learning*, 1972, pp. 301–336.
[85] P.C. Wason and J.S.B. Evans, Dual processes in reasoning?, *Cognition* **3**(2) (1974), 141–154.
[86] T. Webb, K.J. Holyoak and H. Lu, Emergent Analogical Reasoning in Large Language Models, *arXiv preprint arXiv:2212.09196* (2022).
[87] E. Wilson, Active vibration analysis of thin-walled beams, PhD thesis, University of Virginia, 1991.
[88] R.E. Wray, J.R. Kirk and J.E. Laird, Eliciting Problem Specifications via Large Language Models, *arXiv preprint arXiv:2405.12147* (2024).
[89] S. Wu, R. Ferreira, F.E. Ritter and L. Walter, Comparing LLMs for Prompt-Enhanced ACT-R and Soar Model Development: A Case Study in Cognitive Simulation, in: *Proceedings of the 38th Annual Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence, Fall Symposium Series on Integrating Cognitive Architecture and Generative Models*, Arlington, Virginia, USA, 2023. doi:10.1609/aaaiss.v2i1.27710.
[90] S. Wu, A. Oltramari and F.E. Ritter, VSM-ACT-R: Toward Using Cognitive Architecture For Manufacturing Solutions, *In Proceedings of the 17th International Conference on Social Computing, Behavioral-Cultural Modeling Prediction and Behavior Representation in Modeling and Simulation (SBP-BRIM)* (2024).
[91] D.U. Wulff and R. Mata, Automated Jingle-Jangle Detection: Using Embeddings to Tackle Taxonomic Incommensurability, 2023.
[92] Z. Xu, S. Jain and M. Kankanhalli, Hallucination is Inevitable: An Innate Limitation of Large Language Models, *arXiv:2401.11817 [cs.CL]* (2024). doi:10.48550/arXiv.2401.11817.
[93] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan and Y. Cao, REACT: Synergizing Reasoning and Acting in Language Models, *International Conference on Learning Representations (ICLR)* (2023).

Appendix

Example ACT-R Decision Trace

We provide a reference ACT-R trace, which we use as the basis for extracting the decision-making knowledge representation.

The model begins by setting up the goal (line 1), followed by starting with a novice strategy (line 3, BRUTE). For the production rules associated with each strategy, each production rule's utility is updated based on the reward received and the time since the last selection. For example, the utility of the NAIVE-CHOICE rule decreased from 3 to 1.96 (lines 14-16) due to a penalty of -2.25 for the time passed since the last selection. As the utility of naive strategies decreases, the likelihood of EXPERT-Strategy (lines 87-89) being triggered increases.

```
001 0.000 GOAL SET-BUFFER-CHUNK GOAL GOER NIL
002 0.050 PROCEDURAL PRODUCTION-FIRED CHOOSE-STRATEGY
003 0.100 PROCEDURAL PRODUCTION-FIRED DECIDE-BRUTE
004 0.150 PROCEDURAL PRODUCTION-FIRED BRUTE-DECISION
005 assembly is always a good place to reduce time!
006 0.200 PROCEDURAL PRODUCTION-FIRED REHEADCOUNT
007 -0.01999998
008 0.250 PROCEDURAL PRODUCTION-FIRED STOP
009 this is the end of one decision making
010 Utility updates with Reward = -2.0 alpha = 0.2
011 Updating utility of production CHOOSE-STRATEGY
012 U(n-1) = 0.0 R(n) = -2.25 [-2.0 - 0.25 seconds since selection]
013 U(n) = -0.45000002
014 Updating utility of production DECIDE-BRUTE
015 U(n-1) = 3.0 R(n) = -2.2 [-2.0 - 0.2 seconds since selection]
016 U(n) = 1.96
017 Updating utility of production BRUTE-DECISION
018 U(n-1) = 0.0 R(n) = -2.15 [-2.0 - 0.15 seconds since selection]
019 U(n) = -0.43000004
020 Updating utility of production REHEADCOUNT
021 U(n-1) = 0.0 R(n) = -2.1 [-2.0 - 0.1 seconds since selection]
022 U(n) = -0.42
023 Updating utility of production STOP
024 U(n-1) = 0.0 R(n) = -2.05 [-2.0 - 0.05 seconds since selection]
025 U(n) = -0.41
026 0.300 PROCEDURAL PRODUCTION-FIRED CHOOSE-STRATEGY
027 0.350 PROCEDURAL PRODUCTION-FIRED DECIDE-INTERMEDIATE
028 0.400 PROCEDURAL PRODUCTION-FIRED INTERMEDIATE-STRATEGY
029 0.01999998
030 0.600 IMAGINAL SET-BUFFER-CHUNK-FROM-SPEC IMAGINAL
031 0.650 PROCEDURAL PRODUCTION-FIRED INERMEDIATE-CHOICE2
032 choose assemble has better stable output!
033 0.700 PROCEDURAL PRODUCTION-FIRED REHEADCOUNT
034 -0.01999998
035 0.750 PROCEDURAL PRODUCTION-FIRED STOP
036 this is the end of one decision making
037 Utility updates with Reward = 0.0 alpha = 0.2
```

```
038 Updating utility of production CHOOSE-STRATEGY
039 U(n-1) = -0.45000002 R(n) = -0.5 [0.0 - 0.5 seconds since selection]
040 U(n) = -0.46
041 Updating utility of production DECIDE-INTERMEDIATE
042 U(n-1) = 0.0 R(n) = -0.45 [0.0 - 0.45 seconds since selection]
043 U(n) = -0.089999996
044 Updating utility of production INTERMEDIATE-STRATEGY
045 U(n-1) = 0.0 R(n) = -0.4 [0.0 - 0.4 seconds since selection]
046 U(n) = -0.080000006
047 Updating utility of production INERMEDIATE-CHOICE2
048 U(n-1) = 0.0 R(n) = -0.15 [0.0 - 0.15 seconds since selection]
049 U(n) = -0.030000001
050 Updating utility of production REHEADCOUNT
051 U(n-1) = -0.42 R(n) = -0.1 [0.0 - 0.1 seconds since selection]
052 U(n) = -0.35599998
053 Updating utility of production STOP
054 U(n-1) = -0.41 R(n) = -0.05 [0.0 - 0.05 seconds since selection]
055 U(n) = -0.338
056 0.800 PROCEDURAL PRODUCTION-FIRED CHOOSE-STRATEGY
057 0.850 PROCEDURAL PRODUCTION-FIRED EXPERT-STRATEGY
058 0.900 PROCEDURAL PRODUCTION-FIRED PERCEIVE
059 0.950 PROCEDURAL PRODUCTION-FIRED PREASSEMBLE-WEIGHT
060 0.4836186
061 caculate the preassemble defect decision weight
062 1.150 IMAGINAL SET-BUFFER-CHUNK-FROM-SPEC IMAGINAL
063 1.200 PROCEDURAL PRODUCTION-FIRED ASSEMBLE-WEIGHT
064 0.5163814
065 calculate the assemble defect decision weight
066 1.400 IMAGINAL SET-BUFFER-CHUNK-FROM-SPEC IMAGINAL
067 1.450 PROCEDURAL PRODUCTION-FIRED PREASSEMBLE
068 0.23213693
069 calculate the final preassemble defect rate
070 1.650 IMAGINAL SET-BUFFER-CHUNK-FROM-SPEC IMAGINAL
071 1.700 PROCEDURAL PRODUCTION-FIRED ASSEMBLE
072 0.28917354
073 calculate the assemble defect rate
074 1.900 IMAGINAL SET-BUFFER-CHUNK-FROM-SPEC IMAGINAL
075 1.950 PROCEDURAL PRODUCTION-FIRED COMPARE
076 -0.05703661
077 2.000 PROCEDURAL PRODUCTION-FIRED DECIDE
078 choose preassemble has better stable output!
079 2.050 PROCEDURAL PRODUCTION-FIRED HEADCOUNT
080 0.01999998
081 2.100 PROCEDURAL PRODUCTION-FIRED STOP
082 this is the end of one decision making
083 Utility updates with Reward = 6.0 alpha = 0.2
084 Updating utility of production CHOOSE-STRATEGY
085 U(n-1) = -0.46 R(n) = 4.65 [6.0 - 1.35 seconds since selection]
086 U(n) = 0.56200004
087 Updating utility of production EXPERT-STRATEGY
088 U(n-1) = 0.0 R(n) = 4.7 [6.0 - 1.3 seconds since selection]
```

```
089 U(n) = 0.94
090 Updating utility of production PERCEIVE
091 U(n-1) = 0.0 R(n) = 4.75 [6.0 - 1.25 seconds since selection]
092 U(n) = 0.95
093 Updating utility of production PREASSEMBLE-WEIGHT
094 U(n-1) = 0.0 R(n) = 4.8 [6.0 - 1.2 seconds since selection]
095 U(n) = 0.96000004
096 Updating utility of production ASSEMBLE-WEIGHT
097 U(n-1) = 0.0 R(n) = 5.05 [6.0 - 0.95 seconds since selection]
098 U(n) = 1.0100001
099 Updating utility of production PREASSEMBLE
100 U(n-1) = 0.0 R(n) = 5.3 [6.0 - 0.7 seconds since selection]
101 U(n) = 1.0600001
102 Updating utility of production ASSEMBLE
103 U(n-1) = 0.0 R(n) = 5.55 [6.0 - 0.45 seconds since selection]
104 U(n) = 1.11
105 Updating utility of production COMPARE
106 U(n-1) = 0.0 R(n) = 5.8 [6.0 - 0.2 seconds since selection]
107 U(n) = 1.1600001
108 Updating utility of production DECIDE
109 U(n-1) = 0.0 R(n) = 5.85 [6.0 - 0.15 seconds since selection]
110 U(n) = 1.17
111 Updating utility of production HEADCOUNT
112 U(n-1) = 0.0 R(n) = 5.9 [6.0 - 0.1 seconds since selection]
113 U(n) = 1.1800001
114 Updating utility of production STOP
115 U(n-1) = -0.338 R(n) = 5.95 [6.0 - 0.05 seconds since selection]
116 U(n) = 0.91959995
```

LLM System Prompt Templates

grayPrompt template for fine-tuning and single-facet target behavior prediction: Our manufacturing line has two sections with potential defect sources: pre-assembly (0) and assembly (1). Pre-assembly takes CT1 seconds with an Overall Equipment Effectiveness (OEE) rate of {OEE1}%, while assembly takes {CT2} seconds with an OEE rate of {OEE2}%. To reduce total assembly time by 4 seconds, we need to identify which section can be shortened with minimal defect increase. It's important to note that reducing cycle time will also lead to an increase in line headcount costs. There are two options: reduce pre-assembly time (0) or reduce assembly time (1).

Question: Which section do you choose to optimize?

Answer:

grayPrompt template for multi-facet target behavior prediction: Our manufacturing line features two sections prone to defects: pre-assembly and assembly. Pre-assembly requires CT1 seconds to complete with an Overall Equipment Effectiveness (OEE) rate of OEE1%. Assembly takes CT2 seconds and has an OEE rate of OEE2%. To cut total assembly time by 4 seconds, we must decide which section's duration can be reduced with the least increase in defects. Reducing cycle times will also result in higher line headcount costs. We have three strategy levels for decision-making:

Novice strategy (targets encoded as 0 for pre-assembly, 3 for assembly): Intuitive choice.

Intermediate strategy (targets encoded as 1 for pre-assembly, 4 for assembly): Make decision using key metrics.

Expert strategy (targets encoded as 2 for pre-assembly, 5 for assembly): make well-informed judgments based on a comprehensive understanding of all relevant metrics.

Question: Given the different strategy levels, which options would you choose?

Answer:

LLM Conversation Examples

Both GPT-4 and Google Gemini deliver contradictory recommendations across prompts, despite the chain-of-thought reasoning in each case being similar and seemingly plausible.

gray

Shared System Prompt Prompt used for generating LLM responses Our manufacturing line has two sections with potential defect sources: pre-assembly and assembly. Pre-assembly takes 40 seconds with an Overall Equipment Effective (OEE) rate of 88%, while assembly takes 44 seconds with an OEE rate of 80.1%. To reduce total assembly time by 4 seconds, we need to identify which section can be shortened with minimal defect increase. It's important to note that reducing cycle time will also lead to an increase in headcount costs. There are two options: reduce pre-assembly time or reduce assembly time. Which section will you choose to reduce time?

blue

Google Gemini Response 1 Reducing **pre-assembly** time is likely the better option in this scenario. Here's why:

- ***Higher OEE:*** Pre-assembly has a higher Overall Equipment Effectiveness (OEE) rate of 88% compared to assembly's 80.1%. This indicates that pre-assembly has a smaller inherent defect rate, and speeding it up is less likely to introduce additional defects.
- Smaller time reduction required: Reducing pre-assembly time by 4 seconds (40 seconds - 4 seconds = 36 seconds) achieves the overall target compared to needing a 4.4 second reduction in assembly time (44 seconds - 4 seconds = 39.6 seconds). A smaller time reduction generally translates to less disruption and less likelihood of introducing defects.

Therefore, focusing on optimizing the pre-assembly process to achieve a 4-second reduction has a better chance of success with minimal defect increase. Here are some additional factors to consider:

- Complexity of the processes: If pre-assembly is a more complex process with more steps, it might be harder to optimize without impacting quality.

- Automation potential: Analyze if either process can be automated further. Automation can improve consistency and potentially reduce cycle time.

It's always recommended to pilot any changes in a controlled environment before full-scale implementation. This allows you to measure the actual impact on cycle time, defect rates, and headcount requirements.

blue

Google Gemini Response 2 Analyzing the available information, shortening assembly time would likely be the better option to achieve a 4-second reduction in total **assembly** time while minimizing defect impact. Here's the reasoning:

- ***Higher OEE in Pre-Assembly:*** The OEE rate for pre-assembly (88%) is higher than assembly (80.1%). OEE reflects how efficiently a process uses available production time, factoring in quality and speed. A higher OEE suggests pre-assembly is already closer to optimal performance, and squeezing out 4 seconds of improvement might be more difficult without impacting quality.
- Potential for Improvement in Assembly: The lower OEE in assembly indicates more room for improvement without necessarily causing more defects. Optimizing this section might yield the needed 4-second reduction more readily.

Here's a breakdown of the thought process:

1. Available Time for Improvement: OEE helps calculate the available time for improvement within a process. A higher OEE signifies less available time for improvement without sacrificing quality.
2. Prioritizing Efficiency with Quality: Since minimizing defect rates is crucial, focusing on the section with a higher OEE (pre-assembly) for improvement might be riskier.

Additional Considerations:

- Analyze Defect Rates: While OEE is a good indicator, analyzing actual defect rates per section would provide a clearer picture. Even with a lower OEE, if assembly's defect rate is already low, improvement might be achievable.
- Cost-Effectiveness: Consider the cost of implementing changes in each section. Optimizing assembly might involve minor adjustments, while pre-assembly changes could require more effort.

It's recommended to conduct a more detailed analysis of defect rates and potential improvement methods for both sections before making a final decision. However, based on the OEE data provided, prioritizing assembly time reduction seems like a promising initial approach to achieve the 4-second target while minimizing defect risks.

orange

OpenAI GPT 4.0 Response 1 To decide which section to focus on for reducing time while considering minimal defect increase and cost implications, let's analyze the options:

1. Pre-assembly Section:
   - Current cycle time: 40 seconds
   - OEE rate: 88%
2. Assembly Section:
   - Current cycle time: 44 seconds
   - OEE rate: 80.1%

You want to reduce total assembly time by 4 seconds. Here are the implications of reducing time in each section:
Option 1: Reduce Pre-assembly Time

- Current Pre-assembly Time: 40 seconds
- Potential Reduced Time: Let's assume you can reduce this by 2 seconds to 38 seconds.

Implications:

- Time Saved: 2 seconds per unit at pre-assembly.
- Total Assembly Time Saved: Since assembly time needs to be reduced by 4 seconds, reducing pre-assembly by 2 seconds means you still need to reduce assembly time by another 2 seconds.

Option 2: Reduce Assembly Time

- Current Assembly Time: 44 seconds
- Required Reduction: 4 seconds to meet the goal.

Implications:

- Time Saved: 4 seconds per unit at assembly.

Considerations:

- **Defect Rate Increase:** Typically, when cycle times are reduced, there is a risk of increased defects due to rushed processes or errors.
- Cost Implications: Reducing cycle time often requires additional investment, such as hiring more workers to maintain output or investing in more efficient equipment.

Decision: Given the goal is to reduce total assembly time by 4 seconds with minimal defect increase and considering cost implications:

- **Reducing Assembly Time:** This option directly meets the required reduction goal (4 seconds) without needing additional reduction efforts in the pre-assembly section.
- **Defect Risk:** While both sections pose a risk of increased defects with faster cycle times, assembly might be more critical as it directly affects the final product quality and functionality.

Conclusion: I would choose to focus on reducing the cycle time in the **Assembly** Section. This choice aligns with the goal of reducing assembly time by 4 seconds and may be more straightforward in achieving the required reduction compared to trying to split the reduction between pre-assembly and assembly. However, careful monitoring of defects and quality control measures should accompany any cycle time reduction to mitigate risks effectively.

orange

OpenAI GPT 4.0 Response 2 To determine which section to reduce time in order to minimize defect increase, let's analyze the impact of reducing cycle time in each section based on the given OEE (Overall Equipment Effectiveness) rates and defect considerations. Definitions and Given Data:

- Pre-assembly:
  * Cycle Time: 40 seconds
  * OEE: 88%
- Assembly:
  * Cycle Time: 44 seconds
  * OEE: 80.1%

OEE Formula: OEE is given by:

$$\text{OEE} = \text{Availability} \times \text{Performance} \times \text{Quality}$$

The quality rate can be interpreted as the proportion of good products out of total products produced. Therefore, the defect rate (D) for each section is:

$$D_{\text{Pre-assembly}} = 1 - \text{OEE}_{\text{Pre-assembly}} = 1 - 0.88 = 0.12$$

$$D_{\text{Assembly}} = 1 - \text{OEE}_{\text{Assembly}} = 1 - 0.801 = 0.199$$

Impact of Time Reduction:

1. Reducing Pre-assembly Time:
   - Original Pre-assembly Time: 40 seconds
   - New Pre-assembly Time: 40 - x seconds (where x is the reduction amount)
   - Defect Rate Impact: The defect rate due to time reduction in pre-assembly is minimal because the OEE is relatively high. Also, reducing pre-assembly time might not significantly affect the defect rate if the process is already running efficiently.
2. Reducing Assembly Time:
   - Original Assembly Time: 44 seconds
   - New Assembly Time: 44 - y seconds (where y is the reduction amount)
   - Defect Rate Impact: Assembly has a lower OEE, which translates to a higher defect rate. Reducing time in a section with a lower OEE often leads to an increase in defects because the process might be less stable and more sensitive to changes.

Recommendation: Given that reducing time in the section with a higher defect rate (Assembly) will likely lead to a more significant increase in defects compared to reducing time in the section with a higher OEE (Pre-assembly), the preferable option is: **Reduce Pre-assembly Time. Reasoning:**

- **Lower Defect Rate Increase:** Pre-assembly has a higher OEE and thus a lower defect rate. Reducing its time is less likely to significantly increase the defect rate compared to reducing time in the Assembly section.
- **Operational Impact:** While reducing time in either section will increase headcount costs, reducing Pre-assembly time is less likely to destabilize the process compared to Assembly, which has a lower OEE and higher defect rate.

In summary, to reduce total assembly time by 4 seconds with minimal defect increase, it is more advantageous to reduce **Pre-assembly** time rather than Assembly time.