# Comparing LLMs for Prompt-Enhanced ACT-R and Soar Model Development: A Case Study in Cognitive Simulation

Siyu Wu (sfw5621@psu.edu), Rodrigo F. Souza (rodrigo.ferreira@unifesp.br),
Frank E. Ritter (frank.ritter@psu.edu), Walter T. Lima Jr. (walter.lima@unifesp.br)

AAAI FSS

Oct. 25th, 2023

# Nuggets

- Experiments show that large language models (LLMs) have the potential to be used as interactive interfaces to develop ACT-R and Soar models.
- We documented and resolved the mistakes that LLMs made during this integration.
- We also presented a framework of prompt patterns that maximizes LLMs' interaction with artificial cognitive architectures.

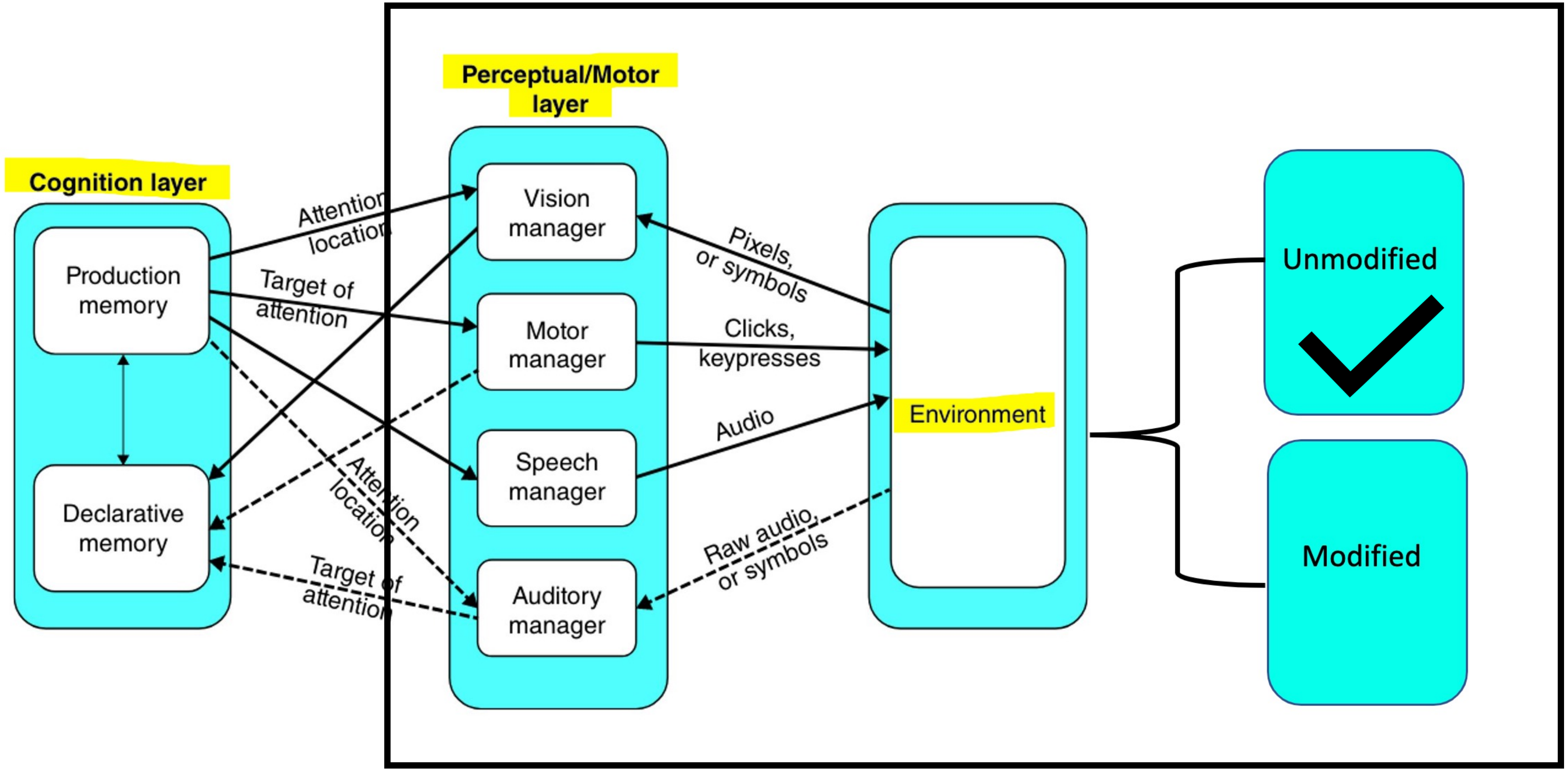
I know you're all also eager to hear about the coal of this study. Just be patient, we'll get there.

# Why use LLMs to build models?

- ACT-R and Soar are powerful frameworks for simulating human behavior.
- However, traditional model development for these frameworks is complex and intimidating.
- LLMs offer new possibilities for enhancing ACT-R and Soar model development, and more research is needed.

Cerf, V. G. (2023). Large Language Models. *Commun. ACM*, *66*(8): 7.

# ACT-R



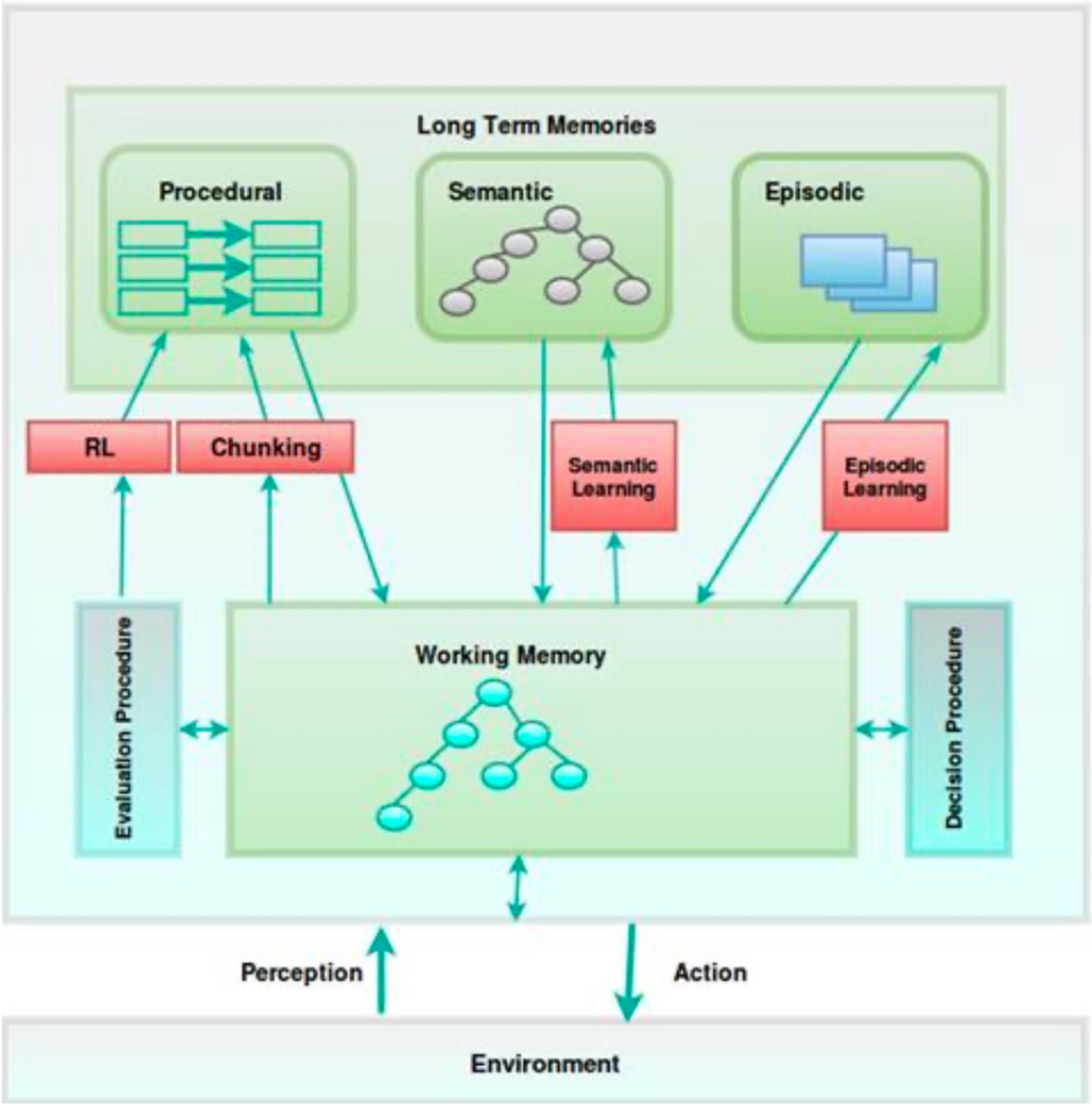
Perceptual/Motor layer
Cognition layer
Production memory
Declarative memory
Attention location
Target of attention
Attention location
Target of attention
Vision manager
Motor manager
Speech manager
Auditory manager
Pixels, or symbols
Clicks, keypresses
Audio
Raw audio, or symbols
Environment
Unmodified
Modified


# Simulation task

- Autonomous driving game on Stream
- Desert Bus

📘 Wu, S., Bagherzadeh, A., Ritter, F. E., & Tehranchi, F. (2023). Cognition models bake-off: Lessons learned from creating long-running cognitive models. In *BRIMS 23*.

# Soar


Long Term Memories
Procedural
Semantic
Episodic
RL
Chunking
Semantic Learning
Episodic Learning
Evaluation Procedure
Working Memory
Decision Procedure
Perception
Action
Environment

# Simulation task

Analyze student responses in a database and identify the dominant intelligence type.

## INTELLIGENCE TYPES:

**Spatial:** Think abstractly

**Bodily-kinesthetic:** Use your body

**Musical:** Sensitivity to rhythm

**Linguistic:** Expose ideas

**Logical-mathematical:** Analyze problems logically

**Interpersonal:** Interact with others

**Intrapersonal:** Sensitivity to one's own feelings

**Naturalistic:** Understand the environment

**Existentialist:** Concrete facts of his life

# Research questions

How effectively can ChatGPT4 and Google Bard, as Large Language Models (LLMs), serve as interactive interfaces for developing ACT-R and Soar models in the context of cognitive task simulation?

What are the patterns of prompt design that maximize LLM outputs to create models in cognitive architectures?

# Theoretical foundations


- Anderson, J. R. (2009). *How can the human mind occur in the physical universe?* New York, NY: Oxford University Press.
- Laird, J. E. (2012). *The Soar cognitive architecture*. MIT Press.
- Newell, A. (1990). Unified Theories of Cognition. *The American Journal of Psychology*, 107(3): 454–464.
- Ritter, F.; Tehranchi, F.; & Oury, J. (2018). ACT-R: A cognitive architecture for modeling cognition. *Wiley Interdisciplinary Reviews: Cognitive Science*, *10*: e1488.

# Prompt design

Providing context

Posing perception-related queries

Simulating decision-making scenarios

Evaluating the system's responses

Iterative fine tuning

# Output optimization

- Sequential steps and actions
- Situational awareness
- Human in the loop

# Model development

- Provide Context and Prompt for Declarative Chunk(s)
- Human In the Loop and Query Initial Rule
- Test and Debug the Code
- Human In the Loop and Query Following Rule
- Model Evaluation
- Duplicates existing model (📘 Wu et al., 2023)

# Provide Context and Prompt for Declarative Chunk(s)

User: You are going to act as an ACT-R modeler, to start with, what you know about ACT-R?

Situational awareness

User: Thank you for your input, here is more info about ACT-R syntax and production rules. Specific info here

Synchronous Domain-Specific Training

User: Now, I am going to give you a simulation task using ACT-R. The simulation environment is the video game Desert Bus VR, which you can find here: [https://store.steampowered.com/app/638110/Desert_Bus_VR/]. The model, named DriveBus, will look around the simulation environment. If the model finds the center lane, it will press the "W" key to start the bus. What are the possible chunk types that you would recommend in this model? What are the possible chunk types that you would recommend in this model?

Provide context

Prompt for Declarative Chunk(s)

# Human In the Loop and Query Initial Rule

- User: Thanks for your reply. Based on your recommendation, the model's name is DriveBus, and it will have one chunk type called drive. This chunk type has two slots: state and visual-cue. The state slot represents the current state or status of the model, such as start, looking around, or driving. The visual-cue slot represents visual cues in the game environment that the model can perceive, such as center lane. Please help to write the ACT-R code, including declarative memories and the first production rule of the model to look around the environment.
- ChatGPT:

Sequential steps and actions

lisp Copy code

```lisp
(clear-all)

;; Chunk type
(chunk-type drive state visual-cue)

;; Declarative memories
(define-chunks
  (drive state start visual-cue center-lane)
)

;; Goal buffer
(goal-focus DriveBus)

;; Production rules
(p look-around
  =goal>
    ISA DriveBus
    state start
==>
  =goal>
    state looking-around
  +visual>
    ISA move-attention
    screen-pos =visual-cue
)
```

# Test and debug the code

Keep the declarative memories generated by ChatGPT, and don't change the model's underlying structure. Focus on syntax.

# Mistakes that LLMs made and the corresponding remedies (by hand)

1. **Problem:** Fail to define the model structure in ACT-r.
   **Resolution**: Include the necessary ACT-R model declaration in the beginning by adding: ”(define-model Drive-Bus …)”.
2. **Problem**: Absence of visual buffer check for initial state.
   **Resolution**: Add the following to the RHS to empty the visual buffer for model initialization: ”+visual-location :attended nil”.
3. **Problem**: Misidentification of chunk type.
   **Resolution**: Change ”isa move-attention” to ”cmd move-attention”.
4. **Problem**: Missing visual buffer LHS check.
   **Resolution**: Add the following to the LHS to meet the requirement of the visual buffer check: ”?Visual >State free”.
5. **Problem**: Miss slot argument.
   **Resolution**: Add ”Center-line” as the slot visual-cue argument when defining the ”drive” chunk.

# Some general mistakes for ChatGPT4-generated ACT-R models that involve perceptual and motor behaviors

- Change ”!eval! (output – key “w”)”, which has no functionality, to:

```
+manual>
  ISA punch
  hand right
  finger index
```

- Add the following lines of code to enable the modules to function properly.

```
(install-device ’(”motor” ”keyboard”))
(add-visicon-features ’(screen-x XX screen-y XX value center-line))
```

# Human In the Loop and Query Following Rule (comparative cases)

- Modeler: We have debugged the previous model and this is the debugged model, based on the debugged model, I need you to help me write one more production rule that if the model see the "center-lane", it will press the "w" key to start the bus.
- Modeler : I have debugged the previous model that you generated, there are several mistakes that you have made in terms of ACT-R syntax and production rules. first, it missed visual buffer LHS check , and I corrected by adding the following to the LHS to meet the requirement of the visual buffer check: ”?Visual >State free. Second…, third... This is the debugged model, now based on the lesson that you have learned for creating the first production rule and the debugged model, I need you to produce the second production rule that if the model see the "center-lane", it will press the "w" key to start the bus.

Synchronous Domain-Specific Training : How to debug

- The second case has superior code quality to the first case, which generates a syntactically correct production rule, while the prompts that contain only corrected code still produce one syntax error: the LHS side should be =visual>, but was mistakenly written as ?visual>. This makes the model unable to meet the requirements of production rule sequential firing.
- It shows that when using ChatGPT to generate production rules in ACT-R, human-in-the-loop combined with synchronous domain-specific training can increase the syntactic quality of the code.

# Evaluate the generated model

```
CL-USER> (run 1)
     0.000   GOAL                   SET-BUFFER-CHUNK GOAL GOER NIL
     0.000   VISION                 SET-BUFFER-CHUNK VISUAL-LOCATION CHUNK0 NIL
     0.000   VISION                 visicon-update
     0.000   PROCEDURAL             CONFLICT-RESOLUTION
     0.050   PROCEDURAL             PRODUCTION-FIRED READY
     0.050   PROCEDURAL             CLEAR-BUFFER VISUAL-LOCATION
     0.050   VISION                 Find-location
     0.050   VISION                 SET-BUFFER-CHUNK VISUAL-LOCATION CHUNK0
     0.050   PROCEDURAL             CONFLICT-RESOLUTION
     0.100   PROCEDURAL             PRODUCTION-FIRED LOOK-AROUND
     0.100   PROCEDURAL             CLEAR-BUFFER VISUAL-LOCATION
     0.100   PROCEDURAL             CLEAR-BUFFER VISUAL
     0.100   PROCEDURAL             CONFLICT-RESOLUTION
     0.185   VISION                 Encoding-complete CHUNK0-1 NIL
     0.185   VISION                 SET-BUFFER-CHUNK VISUAL CHUNK1
     0.185   PROCEDURAL             CONFLICT-RESOLUTION
     0.235   PROCEDURAL             PRODUCTION-FIRED PRESS-W-KEY
Pressing W key
```

(“W” accelerates the bus)

# Evaluate the generated model

- The prompts that we fed into Chat-GPT 4 generated a model that performs the corresponding behavior. For the DMs, the model has the necessary chunk-types and slots to decide the driving state based on the visual cues it perceives. For the production rules, it sequentially fires get ready, then looks around, sees the visual pattern using the visicon, and then uses the manual buffer to press the key. This model has the potential to interact with the unmodified and novel simulation environment, and might be helpful to some modelers.

- Similarly, Tehranchi & Bagherzadehkhorasani (@PSU but presented award-winning HFES paper yesterday across town), are using LLM to extend Simulated eyes and hands (VisiTor), e.g., for Desert Bus and general vision models (text -> task chunks:
  R1: NL task text -> VisiTor;
  R2: -> OCR;
  R3: -> chatGPT;
  R4: -> DMs;
  Herbal-like OR chatGPT carries on with rules to use chunks

- It is worth noting that the initial code was not correct enough to run on its own. This can be contrasted with success stories told about working Java and Visual Basic, where existing programs may be used. However, in this case, the semantics of Soar and ACT-R are more complicated, and there may not be enough worked examples that were used to create these LLMs.

Bagherzadeh, A., & Tehranchi, F. (2023 in press). Automatic Error Model (AEM) for user interface design: A new approach to include errors and error corrections in a cognitive user model. In *Proceedings of Human Factors and Ergonomics Society*. Outstanding student paper.

Paik, J., Kim, J. W., Ritter, F. E., & Reitter, D. (2015). Predicting user performance and learning in human-computer interaction with the Herbal compiler. *ACM Transactions on Computer-Human Interaction, 22*(5), Article 25

# Prompt Patterns that Maximize LLMs Interaction for Artificial Cognitive Architectures: A Framework for Evolving Conversational Excellence

- Initiation and Setting the Context using Persona
- Multi-turn talk within the human-in-the-loop approach
- Synchronous Domain-Specific Training
- Provide Diversified Meta-Communications

# Acknowledgement


- Organizing Committee, FSS2023: Integration of Cognitive Architectures and Generative Models.
- Applied Cognitive Science and Human-Centered Artificial Intelligence Labs, @PSU
- Amirreza Bagherzadehkhorasani’s constant tech and conceptual support.