

000 001 002 003 004 005 006 007 008 009 010 CTM-AI: A BLUEPRINT FOR GENERAL AI INSPIRED BY CONSCIOUSNESS

011 **Anonymous authors**
012 Paper under double-blind review

013 ABSTRACT

014 Despite remarkable advances, today’s AI systems remain narrow in scope, falling
015 short of the flexible, adaptive, and multisensory intelligence that characterizes
016 humans. This gap has fueled longstanding debates about whether AI might one day
017 achieve human-like generality or even consciousness, and whether principles of
018 consciousness can inspire new architectures for general AI. This paper presents an
019 early blueprint for implementing a general AI system based on the Conscious Tur-
020 ing Machine (CTM), a formal machine model of consciousness. The CTM designs
021 an enormous number of powerful processors ranging from specialized experts (e.g.,
022 vision–language models, search engines, APIs) to unspecialized general-purpose
023 learners poised to develop their own expertise. Crucially, for whatever problem
024 must be dealt with, the system need not know in advance which processors hold
025 the relevant expertise; instead, multimodal machine learning methods enable the
026 system to select, integrate, and fuse information across processors. We extend
027 the CTM into a practical framework, the CTM-AI, and demonstrate its utility on
028 diverse tasks including multimodal perception, tool learning with multiple APIs,
029 and multi-turn web agent tasks. Together, this work offers a principled and testable
030 blueprint for general AI inspired by computational models of consciousness.

031 1 INTRODUCTION

032 In recent years, progress toward AI models capable of human-like intelligence has inspired debates
033 regarding whether today’s AI and its future counterparts can one day display human-like levels of
034 consciousness. Flipping the debate, we present a concrete blueprint for general AI based on a formal
035 machine model of consciousness, the Conscious Turing Machine (CTM) (Blum & Blum, 2021; 2022;
036 Liang, 2022). The CTM is a simple and formal model of consciousness inspired by Alan Turing’s
037 model of computation (Turing, 1936) and Bernard Baars’ theater model of consciousness (Baars,
038 1993). Critically different from other cognitive architectures and modern LLM/agentic workflows,
039 the CTM has no central executive - no conductor, no stage director (Blum & Blum, 2023). Instead,
040 the CTM employs a global workspace and distributed competition to integrate the power of an
041 enormous collection of parallel independent cognitive, sensory, motor, and extended processors.
042 When a problem needs to be solved, it becomes globally broadcast to all processors, eliciting help
043 from those who might have the expertise, interest, and resources to tackle the problem, even though
044 their talents and abilities might be unknown to a central executive.

045 **Key contribution.** Despite its potential, the CTM is a concept that remains abstract and theoretical.
046 In this work, we bridge this gap by implementing the formal CTM model as a concrete system called
047 CTM-AI which includes (1) multiple specialized processors operating in parallel, (2) a limited
048 capacity workspace enforcing selective attention via up-tree competition, (3) a global broadcast
049 of information via a down-tree from the workspace to all processors, and (4) the formation of
050 links between relevant processors over time, enabling unconscious communication to integrate their
051 knowledge into higher-order multimodal information. Through continuous interaction feedback, and
052 learning from its external world via sensory inputs, predictions, actuators, and feedback, CTM-AI
053 updates its individual processors, processor links, and multiprocessor integration to improve over
054 time. The CTM-AI model addresses several key limitations of current AI paradigms.

- 055 1. **Modular and decomposable:** Existing monolithic foundation models are centrally computed and
056 structurally fixed, which blocks the update of new skills and processors. CTM-AI is more modular,
057 decomposable, and supports the flexible addition or removal of processors and capabilities. CTM-
058 AI can adapt to task-specific features effortlessly without extra training.

054

055 2. **Free of a central executive:** CTM-AI does not require an orchestrator akin to modern agentic
056 workflows, but rather uses its dynamics to automatically determine the information flow and
057 learning over multiple processors. Therefore, compared with multi-agent workflows that have
058 a fixed workflow defined or learned for specific tasks, CTM-AI is a more general and flexible
059 framework suitable for different tasks.

060 3. **Integrated reasoning and agentic flexibility:** Today’s agentic frameworks still struggle with
061 reasoning over multiple modalities. CTM-AI can carry out multi-step multimodal reasoning
062 across processors (integrating text, vision, tools, and more). A special case recovers the ‘o1-style’
063 single-LLM reasoning when only one processor is active, showing that CTM-AI generalizes
064 LLM reasoning and multimodal multi-agent workflows.

065 **Main results.** To evaluate CTM-AI as a general multisensory and multi-action AI, we present
066 quantitative results that showcase its versatility across a broad range of language modeling, mul-
067 timodal perception, human behavior understanding, and agentic tool use tasks. This wide range
068 of tasks highlights its ability to use external tools, processors, and APIs, integrate and reason over
069 multimodal information, and solve complex multi-step problems. Based on our experiments, we
070 find that CTM-AI can achieve comparable or state-of-the-art performance on multimodal perception
071 tasks (MUStARD for sarcasm detection, URFunny for humor detection, NYCartoon for multimedia
072 analysis), tool learning API-using tasks (StableToolBench), and multi-turn agentic tasks (WebArena).
073 Moreover, our ablation study shows that mechanisms designed inside CTM-AI, including long-
074 term memory, fusion, up-tree competition, and down-tree broadcasting, all contribute to the final
075 improvement. Such experiments prove the value of the architecture and inference mechanism design
076 in CTM-AI.

077 2 RELATED WORK

078 **Consciousness and AI** There have been several directions in building AI systems inspired by
079 human consciousness (Blum & Blum, 2024; Zeng et al., 2024; Zhao et al., 2023); we point the reader
080 to Butlin et al. (2023) for a review. Prior efforts have typically emphasized high-level analogies,
081 such as developing multimodal languages that mirror human multisensory processing (Liang, 2022;
082 Liang et al., 2022b; Lohse et al., 2021; Murray & Wallace, 2011; Nanay, 2018), constructing world
083 models that integrate perception, planning, and action (Nottingham et al., 2023; Singer et al., 2022;
084 Hao et al., 2023; Liu et al., 2024; Prasad et al., 2023), or pursuing reasoning paradigms inspired by
085 human cognition, including robustness (Sun, 1995; Zeng et al., 2023), compositionality (Gupta &
086 Kembhavi, 2023; Wu et al., 2021; Zhou et al., 2022), causality (Halpern, 2000; Liu et al., 2023c; Zang
087 et al., 2023), and transparency (Liang et al., 2020; Mota et al., 2021). CTM-AI differs by directly
088 grounding these inspirations in state-of-the-art reasoning and agentic models, operationalizing them
089 into a concrete, extensible system rather than remaining at the level of abstract analogy.

090 **Large foundation models** The recent wave of large pretrained and generative models such as
091 large language models (Achiam et al., 2023; Brown et al., 2020; Radford et al., 2019; Touvron
092 et al., 2023; Zhang et al., 2022), image generation models (Ramesh et al., 2021; Rombach et al.,
093 2022; Saharia et al., 2022), and multimodal foundation models (Liu et al., 2023a;b; Li et al., 2023;
094 Liang et al., 2022a) have shown emergent abilities across a wide range of tasks (Schaeffer et al.,
095 2023; Wei et al., 2022). Their impressive generalization capabilities have inspired debate on whether
096 these models possess human-level intelligence (Baum et al., 2011) and consciousness (Chalmers,
097 2023). Mixture-of-experts (MoE) has also become a popular design choice for scaling foundation
098 models efficiently. CTM-AI differs fundamentally from large foundation models by moving beyond
099 monolithic scaling with a single centrally trained model and enabling modularity, flexible reasoning,
099 and adaptive agentic behavior. Moreover, CTM-AI differs from multimodal foundation models by
099 designing language-based interaction instead of linear projection for multimodal fusion.

100 **Multi-agent and tool-augmented frameworks** Most multi-agent systems (Qian et al., 2023; Hong
101 et al., 2024; Schmidgall et al., 2025) rely on multi-step prompting pipelines tailored to specific tasks
102 such as coding or reasoning, where each LLM-based agent is assigned a fixed role (e.g., planner,
103 coder, or reviewer). Beyond such task-specific prompting, recent work has focused on enhancing
104 reasoning abilities in LLMs and multimodal models (Li et al., 2025; Dai et al., 2025), reasoning
105 across multiple modalities, extending context and memory (Zhou et al., 2025), and enabling dynamic
106 tool use (Guo et al., 2024; Qin et al., 2023; Yao et al., 2024). While these advances move toward more
107 capable systems, they typically remain tied to fixed role assignments, rigid tool-calling pipelines, or
107 predefined multimodal fusion strategies. In contrast, CTM-AI departs from both directions: unlike

108 multi-agent systems, it is not a task-specific workflow with fixed roles but a general framework
 109 where flexible processors can work together, and unlike tool-augmented LLMs, it does not rely on a
 110 single-step inference but enables adaptive and iterative inference over multiple steps.
 111

112 3 CTM-AI: THE CONSCIOUS TURING MACHINE WITH MODERN AI

113 In this section, we present background on the Conscious Turing Machine (CTM) in §3.1 and explain
 114 how we implement this conceptual model based on modern AI technologies, creating CTM-AI. We
 115 discuss CTM-AI’s core components in §3.2 and its key learning dynamics in §3.3.
 116

117 3.1 BACKGROUND ON THE CONSCIOUS TURING MACHINE (CTM)

118 The CTM is a simple and formal model of consciousness (Blum & Blum, 2021; 2022) inspired by
 119 Alan Turing’s model of computation (Turing, 1936) and Bernard Baars’ theater model of consciousness
 120 (Baars, 1993). However, CTM differs from Turing machines and Baars’ model. While Baars
 121 describes consciousness via the activity of actors performing on a stage directed by a stage director,
 122 the CTM has no stage director or central executive. Designing a central executive can be prohibitive
 123 since we often do not know how such an executive operates. Consider the typical example of trying to
 124 recall the name of a person you’ve previously met. Although we may recall their name eventually, we
 125 do not know which processors are relevant and how to combine processor outputs beforehand. Rather,
 126 a federation of processors runs simultaneously, recalling different locations, events, and memories,
 127 before deciding which outputs are salient and integrating them to form the final answer. Similarly,
 128 the CTM employs a global workspace and distributed competition that determines which information
 129 from its vast collection of “unconscious” cognitive, sensory, and motor processors gets admitted to
 130 the “conscious” arena. When a problem needs to be solved, it becomes globally broadcast to all
 131 processors, eliciting help from those who might have the expertise, interest, and resources to tackle
 132 the problem, even though their talents and abilities might be unknown to a central executive. These
 133 features set the stage for its capability to be a model for general AI (Blum & Blum, 2023).

134 3.2 CTM ARCHITECTURE

135 The formal definition of the CTM is a 7-tuple $\langle \text{STM}, \text{LTM}, \text{Up-Tree}, \text{Down-Tree}, \text{Links}, \text{Input}, \text{Output} \rangle$. We provide a brief explanation for each of them here:
 136

- 137 • CTM is born at time 0 and has a finite lifetime T . Time is measured in discrete clock ticks,
 $t = 0, 1, 2, \dots, T \approx 10^{10}$.
- 138 • STM (short-term memory) is a small memory capable of holding a single chunk of information at
 139 each time t .
- 140 • LTM (long-term memory) is a collection of K powerful processors p_1, p_2, \dots, p_K , K can be as
 141 large as $K > 10^7$.
- 142 • Up-Tree is an up-directed binary tree of height h with K leaves, one leaf in each LTM processor,
 143 and a (single) root in STM.
- 144 • Down-Tree is a simple down-directed tree of height 1 with a single root in STM and K edges
 145 directed from that root to the leaves, one leaf in each LTM processor.
- 146 • Links are the channels for transmitting information directly between processors.
- 147 • Input: $\mathbb{R}^d \rightarrow \text{LTM}$ carries information from the external (outer) world via sensors (e.g., eyes, ears)
 148 to special LTM processors (e.g., visual and auditory processors). \mathbb{R}^d is CTM’s external world
 149 where \mathbb{R} represents the real numbers and d is a positive integer. It also includes a user intent like a
 150 query about the external world.
- 151 • Output: $\text{LTM} \rightarrow \mathbb{R}^d$ carries information from special processors (e.g., motor processor) that can be
 152 considered as feedback to the external (outer) world.

153 **LTM processors.** An LTM processor p_i (with parameters θ_i) operates in a shared space $\mathcal{H} \cong \mathbb{R}^d$ and
 154 maintains a private memory state $M_t \in \mathcal{M}$ updated over time. At step t , it receives an observation
 155 $o_t \in \mathcal{O}$ and a user query $q_t \in \mathcal{Q}$. We view the LTM processor at time t as a function $\text{LTM}_t(\cdot)$ equipped
 156 with three operations: (1) **execute** produces a chunk based on the current observations and previous
 157 memory; (2) **read** returns a view of its memory at a specified timestamp; and (3) **write** integrates one
 158 or more chunks into its memory. Formally:

$$\text{execute: } \text{LTM}_t(o_t, q_t) = \text{chunk}_t \quad (1)$$

$$\text{read: } \text{LTM}_t(\cdot) = M_t \quad (2)$$

$$\text{write: } \text{LTM}_t(\text{chunk}_t^i) = M_t \oplus \text{chunk}_t^i = \text{LTM}_{t+1}(\cdot) \quad (3)$$

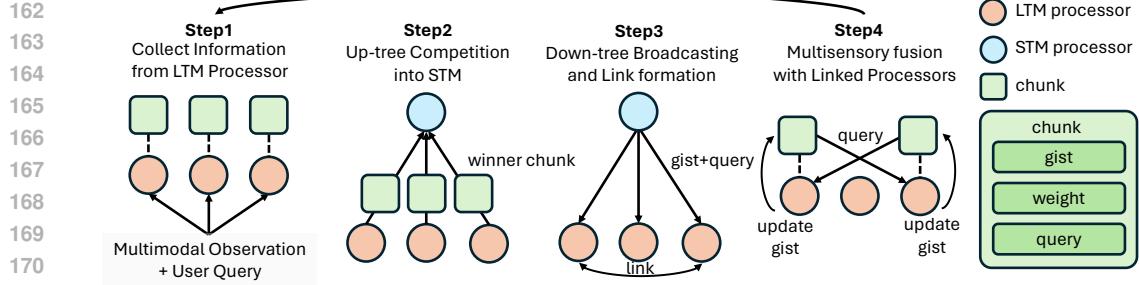


Figure 1: **Key dynamics of CTM-AI:** (1) multiple specialized LTM processors operating in parallel; (2) a limited-capacity STM workspace enforces selective attention via up-tree competition; (3) a global broadcast of information via a down-tree from the workspace to all processors; (4) the formation of links between relevant processors over time, enabling unconscious communication to integrate their knowledge into higher-order multimodal information. CTM-AI continuously interacts with the world through sensing, prediction, action, and feedback, updating its individual processors, processor links, and multiprocessor integration over time.

A chunk c_t^i produced by processor p_i at step t is formally defined as a tuple:

$$\text{chunk}_t^i = \langle \text{addr}(p_i), t, h_t^i, q_t^i, w_t^i \rangle \quad (4)$$

It stores its unique identifier $\text{addr}(p_i)$, the timestep t , a gist $h_t^i \in \mathcal{H}$ in English language that summarizes information relevant to the user’s query (e.g., from audio: “laughter detected; likely humorous”), a follow-up query $q_t^i \in \mathcal{Q}$ that the processor proposes to other processors if answering it could improve the final answer (e.g., ask vision processor for facial expressions), and a weight $w_t^i \in [0, 1]$ indicating the processor’s confidence/utility for how useful the gist is to answering the user’s query. While all LTM processors share the above input–output interface, they differ in their specialties (e.g. input modalities, output tasks, internal memories). Generally, we can group them into five families of LTM processors (Card et al., 1980; Clark & Chalmers, 1998):

- **Sensory processors**, which convert raw perceptual signals such as vision, language, speech, code, music, or more into latent representations.
- **Extended or artificial processors**, which wrap external tools and APIs (e.g., calculators, web search, weather services) so that they can be accessed as internal modules.
- **Cognitive processors**, which handle reasoning and planning over the given query, supporting tasks like commonsense inference or long-horizon problem solving.
- **Motor processors**, which generate outputs by mapping internal intents to external actions, including dialogue utterances, API calls, or embodied movements.
- **Unspecialized “free” processors**, which serve as expandable slots that can acquire new observation, reasoning, or output skills over time through practice and feedback.

STM processors. Besides LTM processors, an STM processor is a stateless LLM (e.g., GPT-4o) that, given the STM at step t that wins the competition among all chunks and the current query q_t , produces a final, text-based answer y_t (i.e., the action a_t) and a quality score $\alpha_t \in [0, 1]$. Unlike LTM processors, it has no long-term memory and therefore only exposes a single **execute** operation; it performs no reads or writes to persistent state and simply grounds its output.

$$\text{execute: } (y_t, \alpha_t) = \text{STM}_t(\text{chunk}_t^*, q_t), \quad \alpha_t \in [0, 1]. \quad (5)$$

3.3 CTM DYNAMICS

Besides the definition of CTM based on a 7-tuple, there are the following dynamics of multisensory processing, information integration, feedback, and learning defined on top of CTM to support its functionality. The overview design principles behind building learning dynamics between multiple processors in the CTM architecture are as below:

1. Different LTM processors perform distinct functions, e.g., cognitive, sensory, or motor. Some processors may be “off-the-shelf” while others’ functionalities are realized over time. While individual processors may have their own internal language, communication within the CTM is in a common multimodal language we call Brainish. All processors start as independent entities.
2. Conscious communication between processors is conducted via an Up-Tree competition that decides whose chunk of information gets into STM.

216 3. The winning chunk (CTM’s conscious content) is immediately globally broadcast to all processors
 217 via the Down-Tree, which causes the CTM to pay conscious attention to this information.
 218 4. Links between processors form over time as one processor views another as having relevant
 219 information, enabling unconscious communication to integrate their knowledge into higher-order
 220 information (e.g., learning to ride a bike requires conscious communication between sight and
 221 movement, after a while, links form, enabling unconscious communication).
 222 5. Through continuous interaction, feedback, and learning from its external world via sensory inputs,
 223 predictions, actuators, and feedback, the CTM updates its individual processors, processor links,
 224 and multiprocessor integration to improve over time.

225 To fully implement such learning dynamics proposed by CTM with modern AI technologies, we split
 226 the overall learning stages of CTM into four parts and provide a more detailed description for each
 227 stage of the learning dynamics as below:

228 **LTM processor chunk inference.** At time t , all LTM processors p_1, \dots, p_K run in parallel on the
 229 observation o_t and query q_t , each using its private memory M_t^i . The collector applies each processor’s
 230 exec to produce chunks (e.g., a VLM on an image, an ALM on audio), yielding:

$$231 \quad \text{CTM}_{\text{collect}}(o_t, q_t) = \{\text{LTM}_t^i(o_t, q_t)\}_{i=1}^K = \{\text{chunk}_t^i\}_{i=1}^K \quad (6)$$

234 **Up-tree competition into STM.** After collecting all chunks from the LTM processors, only one can
 235 be stored in the STM due to its limited capacity. Therefore, an up-tree competition is performed to
 236 select the final chunk. In the original CTM design, this competition is hierarchical and local—each
 237 group of sibling chunks competes using an additive competition function to ensure the probability of
 238 winning is independent of the processor’s position in the tree.

239 However, in our implementation, typically only a few (< 10) LTM processors are active during
 240 inference. Under this setting, restricting competition to local sibling groups is unnecessary, and the
 241 additive function becomes suboptimal. Instead, we adopt a simplified global competition, where the
 242 chunk with the highest score w_t^i (e.g., based on gist quality) is selected as the STM entry:

$$243 \quad \text{CTM}_{\text{up}}(\{\text{chunk}_t^i\}_{i=1}^K) = \text{chunk}_t^{i^*}, \quad i^* := \arg \max_{i \in \{1, \dots, K\}} w_t^i. \quad (7)$$

245 This approach streamlines selection and works well given the small number of competing chunks in
 246 practice. The design can revert to hierarchical selection if large-scale parallelism is introduced.

247 **Down-tree broadcast.** Once the up-tree competition selects the winning chunk $\text{chunk}_t^{i^*}$, it is
 248 written into the STM as $\text{STM}_t^{i^*}$ and immediately broadcast globally to all LTM processors. This
 249 process—called down-tree broadcasting—makes the system consciously aware of this information.
 250 Operationally, each LTM processor receives the broadcast chunk and applies its own **write** function
 251 to update its private memory. This is defined as:

$$253 \quad \text{CTM}_{\text{down}}(\text{chunk}_t^{i^*}) = \{\text{LTM}_t^i(\text{chunk}_t^{i^*})\}_{i=1}^K = \{\text{LTM}_{t+1}^i(\cdot)\}_{i=1}^K \quad (8)$$

255 After this step, the updated memory states are used in the next inference iteration. Conceptually,
 256 this mirrors the system “paying attention” to the winning information at the conscious level and
 257 committing it across all processors for continued reasoning.

258 **Link formation between LTM processors.** Beyond conscious attention, we enable unconscious
 259 communication by dynamically forming links between LTM processors. An unconscious link is
 260 created when one processor identifies another as holding complementary information useful for
 261 improving task performance. For instance, in sarcasm detection, the vision, text, and audio processors
 262 each detect different cues (e.g., sad face, angry tone, exaggerated speech), and over time, they
 263 recognize each other’s utility and form links to exchange information. Concretely, after broadcasting
 264 the winning chunk $\text{chunk}_t^{i^*}$ to processor j , if p_j ’s response yields a high estimated relevance score w_t^j ,
 265 we update the link matrix by adding a small weight increment δ : $L_{i^*j} \leftarrow L_{i^*j} + \delta$ and $L_{ji^*} \leftarrow L_{ji^*} + \delta$.
 266 This mechanism ensures efficient, dynamic linking for cooperative inference. To prevent interference
 267 or propagation of contradictory information, links are not permanent: their linking weights can decay
 268 or be reduced by δ , effectively removing weak or unhelpful connections over time.

269 **Multimodal fusion to update LTM_i.** After down-tree broadcast and link formation, each processor
 LTM_i has a record $\mathcal{N}(i)$ of linked processors $\{\text{LTM}_j\}_{j=1}^M$ that are linked useful for further reasoning.

270 These links support unconscious information exchange. As part of the fusion process, each processor
 271 first generates a query q_{t+1}^i based on its current long-term memory M_{t+1}^i , including the newly
 272 broadcast chunk. Each processor then consults its neighbors in $\mathcal{N}(i)$ in parallel, posing its query q_{t+1}^i
 273 to them. The neighbors respond by running their **execute** function, and the initiating processor uses
 274 their responses to update its memory via the **write** function. This overall process is defined as:

$$275 \quad \text{CTM}_{\text{fuse}}(o_t) = \left\{ \text{LTM}_{t+1}^i \left(\left\{ \text{LTM}_{t+1}^j(o_t, q_{t+1}^i) \right\}_{j \in \mathcal{N}(i)} \right) \right\}_{i=1}^K = \left\{ \text{LTM}_{t+2}^i(\cdot) \right\}_{i=1}^K \quad (9)$$

276 Multisensory integration enables the discovery of richer, higher-order redundant, unique, or syn-
 277 ergistic information from linked processors (Liang et al., 2022b; 2023; Partan & Marler, 1999;
 278 2005).

279 **Overall: prediction, feedback, and learning.** The CTM-AI system operates through an iterative
 280 cycle of prediction, feedback, and learning.

- 281 **Prediction.** The overall prediction phase is described as $\text{CTM}_{\text{up}}(\text{CTM}_{\text{collect}}(o_t, q_t))$. For each
 282 user query q_t , the system collects chunks from LTM processors via $\text{CTM}_{\text{collect}}$, conducts up-
 283 tree competition via $\text{CTM}_{\text{up-tree}}$. This winning chunk that fits into the STM is considered the
 284 prediction provided by CTM.
- 285 **Feedback.** The STM processor (an LLM) evaluates the prediction and assigns a quality score α_t .
 286 If $\alpha_t \geq \tau$, the prediction is accepted as output. Otherwise, negative feedback triggers learning,
 287 prompting the system to refine its internal state before reattempting inference.
- 288 **Learning.** Learning is implemented via in-context learning with memory updates. It consists of two
 289 parts: (1) Down-tree broadcast: The winning chunk is written into each LTM processor’s memory
 290 via CTM_{down} ; (2) Multimodal fusion: Each processor generates a follow-up query, consults its
 291 linked neighbors $\mathcal{N}(i)$, and fuses the resulting information via CTM_{fuse} , enriching its LTM. These
 292 updates prepare the system for improved reasoning in the next iteration.

293 This *prediction–feedback–learning* loop generalizes multiple recent AI innovations, including multi-
 294 step reasoning, agentic workflows, multimodal interaction, and tool use. Crucially, unlike orchestrator-
 295 based systems, CTM-AI does not rely on an external controller. Instead, its intrinsic dynamics—up-
 296 tree selection, down-tree broadcasting, and processor linking—autonomously regulate information
 297 flow and drive continual learning across iterations.

300 4 EVALUATING THE CAPABILITIES OF CTM-AI

301 In this section, we present quantitative results that showcase CTM-AI’s versatility across a broad
 302 range of tasks, including language modeling, multimodal perception, tool use, and agentic tasks. This
 303 wide range of tasks highlights its potential ability to serve as a general AI framework.

304 4.1 EVALUATION TASKS

305 To assess the generality of CTM-AI, we select tasks that *activate distinct subsets of processors*
 306 within each : (i) multimodal grounding and perception (text–audio–image–video); (ii) abstract/social
 307 understanding (affect, humor, sarcasm); (iii) temporal reasoning; (iv) tool use and actuation (planning
 308 API calls and reading/writing external state); and (v) interactive, long-horizon agency (goal decom-
 309 position, feedback handling, recovery from errors). These axes require CTM-AI to *compose perception,*
 310 *cognitive, tool, and agentic processors* iteratively to complete end-to-end tasks.

311 **Multimodal perception.** Real data combine and conflict across modalities (e.g., words vs. tone
 312 vs. visuals). We use MULTIBENCH (Liang et al., 2021) and HEMM (Liang et al., 2024) for broad
 313 modality coverage, plus MUSTARD (Castro et al., 2019), UR-FUNNY (Hasan et al., 2019), and
 314 NYCARTOON (Hessel et al., 2023) for socially grounded semantics (sarcasm, humor, cultural
 315 references). These tasks primarily engage audio/video/text perception processors and cognitive
 316 processors for cross-modal reasoning.

317 **Tool learning.** General systems must not only perceive but also *act*. STABLETOOLBENCH (Guo
 318 et al., 2025) evaluates planning, argument construction, multi-tool composition, and error recovery.
 319 These tasks chiefly engage multiple tool processors (typed API connectors with schema/argument
 320 grounding) to accomplish one task.

321 **Agentic tasks.** Autonomy requires long-horizon control and robustness to stochastic interfaces.
 322 WebArena (Zhou et al., 2023) probes end-to-end web interaction: parsing noisy pages, tracking state,

MUSARD				
Model	Acc↑	P↑	R↑	F1↑
LMF	–	70.73	70.90	70.68
LF-DNN-v1	–	71.55	71.52	70.99
ALBEF	54.49	47.08	50.22	48.51
BLIP2	53.75	48.46	90.13	62.65
MMoE	70.41	60.64	89.04	71.78
BaseModel	70.42	70.44	70.90	70.26
CTM-AI	73.88	73.96	74.44	73.77

Table 1: **CTM-AI evaluation results on MUSARD.**

CTM-AI is able to reach state-of-the-art results on sarcasm detection, beating the base model a lot.

URFunny				
Model	Acc↑	P↑	R↑	F1↑
MulT	66.65	–	–	–
FDMER	71.87	–	–	–
ALBEF	66.77	64.29	73.74	68.67
BLIP2	70.43	65.14	86.60	74.31
MMoE	71.88	69.18	78.16	73.29
BaseModel	60.69	60.77	60.73	60.66
CTM-AI	71.64	71.63	71.62	71.62

Table 2: **CTM-AI evaluation results on URFunny.**

CTM-AI is able to reach comparable results with state-of-the-art models on humor detection.

and replanning. These tasks engage agentic web processors—DOM parser, screenshot/OCR, and AXTree handlers—together with cognitive processors to conduct multi-turn learning and close the perception–planning–action loop.

4.2 BASELINE SETTINGS

Backbone model. To support evaluation across multimodal perception, tool use, and agentic tasks, we adopt `gemini-2.0-flash-lite` as the base model. It natively accepts text, audio, and vision inputs and supports function calling, allowing CTM-AI to expose these capabilities as *processors* (multimodal models and tool callers) within a unified architecture.

Multimodal perception baselines. For MUSTARD and UR-FUNNY, we compare against strong multimodal baselines including MMoE (Yu et al., 2023), BLIP-2 (Li et al., 2023), and ALBEF (Li et al., 2021), which jointly process text and images and report competitive performance on cross-modal understanding.

Tool-using baselines. To assess tool-use competence on STABLETOOLBENCH, we include GPT-4o (Achiam et al., 2023) and ToolLLaMA v2 (Qin et al., 2023) with standard prompting strategies (Chain-of-Thought and DFS-style planning). These systems exhibit strong function-calling ability, composing multiple tools to complete multi-step tasks.

Agentic baselines. For web-based agentic evaluation, we use GPT-4o as a baseline agent. Both the CTM-AI-based agent and the GPT-4o agent receive identical observations (DOM tree, screenshots, and AXTree) and follow a ReAct-style loop (Yao et al., 2023), ensuring a fair comparison of planning and interaction capabilities.

4.3 MAIN RESULTS

CTM-AI achieves state-of-the-art or competitive results across multimodal, tool calling, and agentic benchmarks. As shown in Table 1 and Table 3, CTM-AI attains state-of-the-art performance, improving by *3 points* on multimodal sarcasm detection and by *6+ points* on the tool-calling benchmark. On UR-FUNNY, CTM-AI delivers performance comparable to strong baselines. These settings require non-trivial coordination across processors (e.g., audio–video–text fusion for perception; planning and execution for tools), underscoring CTM-AI’s ability to function as a general AI framework that composes multiple capabilities. Additionally, on a random sample of 40 ONESTOPSHOP cases from WebArena, CTM-AI surpasses LLM-only baselines (CTM-AI succeeds 8 tasks while baseline models only succeed 6) by leveraging its web-agent processors (DOM parsing, screenshot/OCR, AXTree handling) alongside planning and state tracking.

Model performance gains stem from CTM-AI’s mechanisms rather than base-model scaling. Our improvements arise from CTM-AI’s processor orchestration—not from a stronger underlying base model. Built atop the same base model, CTM-AI introduces structured interaction mechanisms (up-tree / down-tree message passing, cross-modal fusion, and link formation) that route information among processors. In multimodal perception, this enables precise cross-modal alignment; in tool use, it captures sequential dependencies and argument grounding across multi-step calls. The same base model, when equipped with CTM-AI’s interaction layer, can thus activate *different subsets of processors* to solve heterogeneous tasks without retraining.

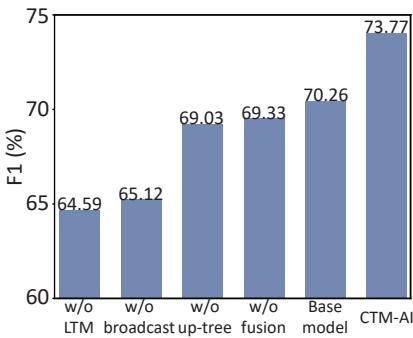


Figure 2: **Ablation study on each mechanism of CTM-AI dynamics.** Full results are available in Appendix §E.

StableToolBench			
Method	I2-Cat.	I2-Inst.	I3-Inst.
ToolLLaMA v2 CoT	19.9±1.0	22.3±0.4	19.1±0.8
ToolLLaMA v2 DFS	22.8±1.5	19.2±1.6	18.6±1.5
GPT-4o mini CoT	24.5±1.0	22.3±2.7	20.8±1.5
GPT-4o mini DFS	25.8±1.7	25.8±2.7	20.2±0.8
GPT-4o CoT	32.5±1.7	29.6±1.6	27.9±3.5
GPT-4o DFS	32.8±1.5	28.3±1.3	23.0±1.3
Base Model+CoT	26.3±1.2	37.2±2.1	18.5±0.9
CTM-AI	39.1±2.0	51.5±1.9	38.5±1.3

Table 3: **CTM-AI evaluation results on StableToolBench.** We refer to the MirrorAPI-Cache. Solvable Pass Rate Score evaluated with GPT-4o. CTM-AI improves the performance a lot.

CTM-AI adapts to diverse real-world tasks with minimal adjustment. Because CTM-AI is made up of multiple processors and modular, it can be ported to *new* tasks by changing only light prompts and routing (i.e., selecting the relevant processor subset) rather than retraining. In practice, adapting from multimodal perception to tool-use or web-based agentic tasks amounts to swapping/adding processors (e.g., a search engine or calculator tool, a DOM/OCR/AXTree stack) and updating task instructions for the same processor; the base model and interaction layer remain unchanged. This plug-and-play design lets CTM-AI meet diverse task requirements with minimal overhead while preserving performance and stability.

4.4 ABLATION STUDIES

Ablation on dynamic mechanisms of CTM-AI. Motivated by the cognitive theory behind CTM-AI, we instantiate CTM dynamics with four key mechanisms: (i) *chunk inference*, (ii) *up-tree competition*, (iii) *down-tree broadcast*, (iv) *link formation*, and (v) *multimodal fusion*. To isolate their contributions, we run comprehensive ablations that selectively disable or replace each mechanism and measure the resulting performance deltas across tasks. As shown in Figure 2, each component plays a non-trivial role and contributes to the overall reasoning ability of CTM-AI, with performance consistently degrading when any of them is removed. The existence of long-term memory is the most important part in the CTM dynamics.

Ablation on Processors of CTM-AI				
Method	Acc↑	P↑	R↑	F1↑
Base model (Only text input)	56.17	61.85	59.66	55.14
CTM-AI (Only language processor)	<u>69.66</u>	<u>69.57</u>	<u>67.41</u>	<u>67.59</u>
Base model (Only audio input)	64.40	67.98	59.77	57.11
CTM-AI (Only audio processor)	<u>67.89</u>	<u>68.11</u>	<u>65.14</u>	<u>65.06</u>
Base model(Only video modality)	61.79	60.59	60.04	60.07
CTM-AI (Only video processor)	58.43	60.37	51.82	42.35
Base model (All modalities combined)	70.42	70.44	70.90	70.26
CTM-AI	73.88	73.96	74.44	73.77

Table 4: **Ablation on single-modality inputs.** When restricted to audio-only or text-only inputs, CTM-AI still outperforms the base model by leveraging broadcasting and unconscious link formation to reason more deeply with limited information.

Ablation on single modality inputs. In Figure 2, we present the ablation results when only a single modality is provided as input, comparing the Base Model (Gemini-2.0-flash-lite) with CTM-AI. The results show that when restricted to audio-only or text-only inputs, CTM-AI consistently outperforms the base model. We argue that this improvement is caused by the broadcasting mechanism and unconscious link formation within CTM-AI, even with limited information, processors can generate follow-up questions that the original modality-specific processor may not have considered. This allows the system to continue reasoning iteratively and explore the input from different perspectives, leading to deeper inference. However, when only the video modality is available,

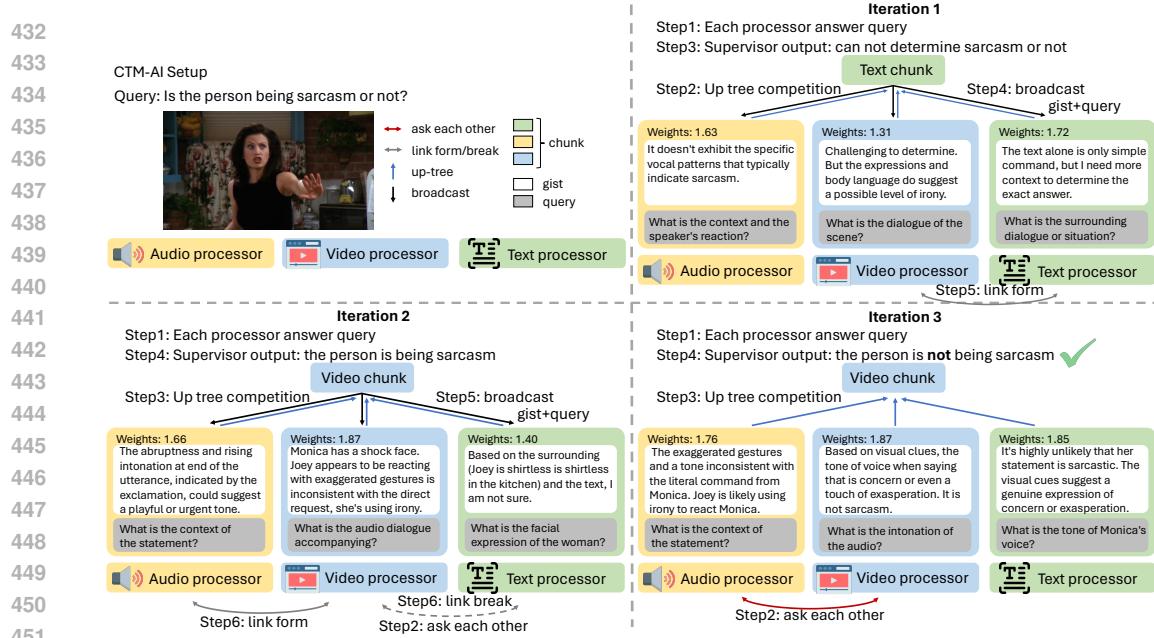


Figure 3: Case study of CTM-AI dynamics. We show three iterations of CTM-AI for sarcasm detection. Through multiple rounds of structured interaction, the system progressively integrates multimodal cues and convergence on the correct interpretation.

CTM-AI performs worse than the base model. We hypothesize that visual information alone can sometimes be misleading, causing the system to propagate incorrect cues through broadcasting without information from other modalities to correct them. Overall, these findings underscore the importance of multi-processor collaboration, when all modalities are jointly available, the system benefits from richer cross-modal interactions, and the performance improves significantly compared to any single-modality setting.

5 CASE STUDY

Based on Figure 3, we analyze a multimodal perception case for identifying *sarcasm*. In the **first** iteration, all three processors are initially *uncertain* about their judgments. The text processor wins the competition, broadcasts its partial understanding of the task to the other processors, and ask explicitly for more context from other processors. This broadcast enables the video processor to respond with relevant visual cues, forming a link of shared information. In the **second** iteration, the video and text processor did unconscious communication with each other. The video processor integrates the contextual cues from the text and its own vision frames, and infers that the speaker is likely being sarcastic, but it still asks for the accompanying audio for a more comprehensive answer. In the **third** iteration, the video processor further queries the audio processor, receiving prosodic and tonal cues. With this enriched multimodal evidence, it refines the judgment and concludes that the speaker is not sarcastic, but instead expressing genuine concern with a shocked and somewhat exaggerated facial expression. Through repeated broadcasting and mutual asking, the processors progressively link their evidence, fuse perspectives, and converge on the correct answer.

6 CONCLUSION

Our work bridges the Conscious Turing Machine (CTM) theory with practical AI by implementing a system that integrates a large number of distributed processors and operates through an iterative prediction–feedback–learning loop. Experiments demonstrate that CTM-AI achieves strong and versatile performance across multimodal perception, tool use, and agentic tasks. Moreover, the architecture adapts to new tasks with minimal adjustment and without retraining. We present CTM-AI as a prototype that connects consciousness theory with general AI, offering a promising foundation for future development.

486 REPRODUCIBILITY STATEMENT
487488 The datasets used in our experiments, MUStARD, URFunny, NYCartoon, StableToolBench and
489 WebArena, are publicly available. Details of test datasets are provided in Section 4 and the imple-
490 mentation of CTM-AI are provided in Appendix D.491 ETHICS STATEMENT
492493 This work builds upon publicly available datasets, no private or sensitive user data were collected or
494 used in this research, and all experiments were conducted in controlled research settings.495 Our research provides a concrete implementation that bridges the theoretical framework of CTM
496 with practical AI technologies. The goal is to enhance LLMs' capabilities in affective learning,
497 decision-making, multi-step reasoning, and tool use, thereby contributing to the development of more
498 reliable and trustworthy general AI systems. Importantly, our intention is not to replicate human
499 identity or create systems indistinguishable from humans, thereby avoiding potential ethical risks
500 associated with anthropomorphization (Deshpande et al., 2023).501 We also recognize the inherent risks of applying large language models and AI agents. These risks
502 include biases that may arise from cultural or social factors. We are committed to ongoing analysis
503 aimed at detecting, understanding, and mitigating such biases. Addressing these challenges remains
504 central to our ethical research framework.505 REFERENCES
506507 Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman,
508 Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report.
509 *arXiv preprint arXiv:2303.08774*, 2023.510 Bernard J Baars. *A cognitive theory of consciousness*. Cambridge University Press, 1993.511 Seth D Baum, Ben Goertzel, and Ted G Goertzel. How long until human-level ai? results from an
512 expert assessment. *Technological Forecasting and Social Change*, 78(1):185–195, 2011.513 Lenore Blum and Manuel Blum. A theory of consciousness from a theoretical computer science
514 perspective: Insights from the conscious turing machine. *Proceedings of the National Academy of
515 Sciences*, 119(21):e2115934119, 2022.516 Lenore Blum and Manuel Blum. A theoretical computer science perspective on consciousness and
517 artificial general intelligence. *Engineering*, 2023.518 Lenore Blum and Manuel Blum. Ai consciousness is inevitable: a theoretical computer science
519 perspective. *arXiv preprint arXiv:2403.17101*, 2024.520 Manuel Blum and Lenore Blum. A theoretical computer science perspective on consciousness.
521 *Journal of Artificial Intelligence and Consciousness*, 8(01):1–42, 2021.522 Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal,
523 Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are
524 few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.525 Patrick Butlin, Robert Long, Eric Elmoznino, Yoshua Bengio, Jonathan Birch, Axel Constant, George
526 Deane, Stephen M Fleming, Chris Frith, Xu Ji, et al. Consciousness in artificial intelligence:
527 insights from the science of consciousness. *arXiv preprint arXiv:2308.08708*, 2023.528 Stuart K Card, Thomas P Moran, and Allen Newell. The keystroke-level model for user performance
529 time with interactive systems. *Communications of the ACM*, 23(7):396–410, 1980.530 Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea,
531 and Soujanya Poria. Towards multimodal sarcasm detection (an `_obviously_` perfect paper). In
532 *ACL*, pp. 4619–4629, 2019.

533 David J. Chalmers. Could a large language model be conscious?, 2023.

534 Andy Clark and David Chalmers. The extended mind. *analysis*, 58(1):7–19, 1998.

540 Wei Dai, Peilin Chen, Chanakya Ekbote, and Paul Pu Liang. Qoq-med: Building multimodal clinical
 541 foundation models with domain-aware grp0 training. *arXiv preprint arXiv:2506.00711*, 2025.

542

543 Ameet Deshpande, Tanmay Rajpurohit, Karthik Narasimhan, and Ashwin Kalyan. Anthropomor-
 544 phization of ai: opportunities and risks. *arXiv preprint arXiv:2305.14784*, 2023.

545 Ning Ding, Sheng-wei Tian, and Long Yu. A multimodal fusion method for sarcasm detection based
 546 on late fusion. *Multimedia Tools Appl.*, 81(6):8597–8616, March 2022. ISSN 1380-7501. doi: 10.
 547 1007/s11042-022-12122-9. URL <https://doi.org/10.1007/s11042-022-12122-9>.

548

549 Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong
 550 Sun, and Yang Liu. Stabletoolbench: Towards stable large-scale benchmarking on tool learning of
 551 large language models. *arXiv preprint arXiv:2403.07714*, 2024.

552

553 Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong
 554 Sun, and Yang Liu. Stabletoolbench: Towards stable large-scale benchmarking on tool learning of
 555 large language models, 2025. URL <https://arxiv.org/abs/2403.07714>.

556

557 Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning
 558 without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern
 559 Recognition*, pp. 14953–14962, 2023.

560

561 Joseph Y Halpern. Axiomatizing causal reasoning. *Journal of Artificial Intelligence Research*, 12:
 562 317–337, 2000.

563

564 Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu.
 565 Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*,
 566 2023.

567

568 Md Kamrul Hasan, Wasifur Rahman, AmirAli Bagher Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer,
 569 Louis-Philippe Morency, and Mohammed Ehsan Hoque. Ur-funny: A multimodal language dataset
 570 for understanding humor. In *Proceedings of the 2019 Conference on Empirical Methods in Natural
 571 Language Processing and the 9th International Joint Conference on Natural Language Processing
 572 (EMNLP-IJCNLP)*, pp. 2046–2056, 2019.

573

574 Jack Hessel, Ana Marasović, Jena D. Hwang, Lillian Lee, Jeff Da, Rowan Zellers, Robert Mankoff,
 575 and Yejin Choi. Do androids laugh at electric sheep? humor "understanding" benchmarks from the
 576 new yorker caption contest, 2023. URL <https://arxiv.org/abs/2209.06293>.

577

578 Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin
 579 Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for
 580 a multi-agent collaborative framework. International Conference on Learning Representations,
 581 ICLR, 2024.

582

583 Hengzhi Li, Brendon Jiang, Alexander Naehu, Regan Song, Justin Zhang, Megan Tjandrasuwita,
 584 Chanakya Ekbote, Steven-Shine Chen, Adithya Balachandran, Wei Dai, et al. Puzzleworld: A
 585 benchmark for multimodal, open-ended reasoning in puzzlehunts. *arXiv preprint arXiv:2506.06211*,
 586 2025.

587

588 Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven
 589 Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum
 590 distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

591

592 Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image
 593 pre-training with frozen image encoders and large language models. In *International conference
 594 on machine learning*, pp. 19730–19742. PMLR, 2023.

595

596 Paul Pu Liang. Brainish: Formalizing a multimodal language for intelligence and consciousness.
 597 *Annual Meeting of the Association for the Scientific Study of Consciousness*, 2022.

598

599 Paul Pu Liang, Yiwei Lyu, Xiang Fan, Zetian Wu, Yun Cheng, Jason Wu, Leslie Yufan Chen,
 600 Peter Wu, Michelle A Lee, Yuke Zhu, et al. Multibench: Multiscale benchmarks for multimodal
 601 representation learning. In *Thirty-fifth Conference on Neural Information Processing Systems
 602 Datasets and Benchmarks Track (Round 1)*, 2021.

594 Paul Pu Liang, Yiwei Lyu, Xiang Fan, Jeffrey Tsaw, Yudong Liu, Shentong Mo, Dani Yogatama,
 595 Louis-Philippe Morency, and Russ Salakhutdinov. High-modality multimodal transformer: Quanti-
 596 fying modality & interaction heterogeneity for high-modality representation learning. *Transactions*
 597 *on Machine Learning Research*, 2022a.

598 Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. Foundations and recent trends in
 599 multimodal machine learning: Principles, challenges, and open questions. *arXiv preprint*
 600 *arXiv:2209.03430*, 2022b.

601 Paul Pu Liang, Yun Cheng, Xiang Fan, Chun Kai Ling, Suzanne Nie, Richard J Chen, Zihao Deng,
 602 Nicholas Allen, Randy Auerbach, Faisal Mahmood, et al. Quantifying & modeling multimodal
 603 interactions: An information decomposition framework. In *Thirty-seventh Conference on Neural*
 604 *Information Processing Systems*, 2023.

605 Paul Pu Liang, Akshay Goindani, Talha Chafekar, Leena Mathur, Haofei Yu, Ruslan Salakhutdinov,
 606 and Louis-Philippe Morency. Hemm: Holistic evaluation of multimodal foundation models.
 607 *Advances in Neural Information Processing Systems*, 37:42899–42940, 2024.

608 Weixin Liang, Feiyang Niu, Aishwarya Reganti, Govind Thattai, and Gokhan Tur. Lrta: A transparent
 609 neural-symbolic reasoning framework with modular supervision for visual question answering.
 610 *arXiv preprint arXiv:2011.10731*, 2020.

611 Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction
 612 tuning, 2023a.

613 Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*,
 614 2023b.

615 Yang Liu, Guanbin Li, and Liang Lin. Cross-modal causal relational reasoning for event-level visual
 616 question answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023c.

617 Yuchen Liu, Luigi Palmieri, Sebastian Koch, Ilche Georgievski, and Marco Aiello. Delta: De-
 618 composed efficient long-term robot task planning using large language models. *arXiv preprint*
 619 *arXiv:2404.03275*, 2024.

620 Michael Lohse, Johannes C Dahmen, Victoria M Bajo, and Andrew J King. Subcortical circuits
 621 mediate communication between primary sensory cortical areas in mice. *Nature Communications*,
 622 12(1):1–14, 2021.

623 Tiago Mota, Mohan Sridharan, and Aleš Leonardis. Integrated commonsense reasoning and deep
 624 learning for transparent decision making in robotics. *SN Computer Science*, 2(4):242, 2021.

625 Micah M Murray and Mark T Wallace. *The neural bases of multisensory processes*. CRC Press,
 626 2011.

627 Bence Nanay. Multimodal mental imagery. *Cortex*, 105:125–134, 2018.

628 Kolby Nottingham, Prithviraj Ammanabrolu, Alane Suhr, Yejin Choi, Hannaneh Hajishirzi, Sameer
 629 Singh, and Roy Fox. Do embodied agents dream of pixelated sheep: Embodied decision making
 630 using language guided world modelling. In *International Conference on Machine Learning*, pp.
 631 26311–26325. PMLR, 2023.

632 Sarah Partan and Peter Marler. Communication goes multimodal. *Science*, 283(5406):1272–1273,
 633 1999.

634 Sarah R Partan and Peter Marler. Issues in the classification of multimodal communication signals.
 635 *The American Naturalist*, 166(2):231–245, 2005.

636 Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal,
 637 and Tushar Khot. Adapt: As-needed decomposition and planning with language models. *arXiv*
 638 *preprint arXiv:2311.05772*, 2023.

639 Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen,
 640 Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. *arXiv*
 641 *preprint arXiv:2307.07924*, 2023.

648 Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru
 649 Tang, Bill Qian, Sihan Zhao, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li,
 650 Zhiyuan Liu, and Maosong Sun. Toollm: Facilitating large language models to master 16000+
 651 real-world apis, 2023.

652 Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language
 653 models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

654 Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen,
 655 and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine
 656 learning*, pp. 8821–8831. Pmlr, 2021.

657 Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-
 658 resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF confer-
 659 ence on computer vision and pattern recognition*, pp. 10684–10695, 2022.

660 Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar
 661 Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic
 662 text-to-image diffusion models with deep language understanding. *Advances in neural information
 663 processing systems*, 35:36479–36494, 2022.

664 Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language
 665 models a mirage? In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.),
 666 *Advances in Neural Information Processing Systems*, volume 36, pp. 55565–55581. Curran Asso-
 667 ciates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/adc98a266f45005c403b8311ca7e8bd7-Paper-Conference.pdf.

668 Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu,
 669 Michael Moor, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using llm agents as research
 670 assistants. *arXiv preprint arXiv:2501.04227*, 2025.

671 Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry
 672 Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video
 673 data. *arXiv preprint arXiv:2209.14792*, 2022.

674 Ron Sun. Robust reasoning: integrating rule-based and similarity-based reasoning. *Artificial
 675 Intelligence*, 75(2):241–295, 1995.

676 Hao Tan and Mohit Bansal. LXMERT: learning cross-modality encoder representations from
 677 transformers. *CoRR*, abs/1908.07490, 2019. URL <http://arxiv.org/abs/1908.07490>.

678 Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée
 679 Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and
 680 efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

681 Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and
 682 Ruslan Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences.
 683 In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual
 684 Meeting of the Association for Computational Linguistics*, pp. 6558–6569, Florence, Italy, July
 685 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1656. URL <https://aclanthology.org/P19-1656/>.

686 Alan Turing. On computable numbers, with an application to the entscheidungsproblem. *J. of Math.*,
 687 58(345–363):5, 1936.

688 Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama,
 689 Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models.
 690 *arXiv preprint arXiv:2206.07682*, 2022.

691 Zhengxuan Wu, Elisa Kreiss, Desmond C Ong, and Christopher Potts. Reascan: Compositional
 692 reasoning in language grounding. *arXiv preprint arXiv:2109.08994*, 2021.

702 Dingkang Yang, Shuai Huang, Haopeng Kuang, Yangtao Du, and Lihua Zhang. Disentangled
 703 representation learning for multimodal emotion recognition. In *Proceedings of the 30th ACM*
 704 *International Conference on Multimedia*, MM '22, pp. 1642–1651, New York, NY, USA, 2022.
 705 Association for Computing Machinery. ISBN 9781450392037. doi: 10.1145/3503161.3547754.
 706 URL <https://doi.org/10.1145/3503161.3547754>.

707 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao.
 708 React: Synergizing reasoning and acting in language models. In *International Conference on*
 709 *Learning Representations (ICLR)*, 2023.

710 Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. tau-bench: A benchmark for
 711 tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

712 Haofei Yu, Zhengyang Qi, Lawrence Jang, Ruslan Salakhutdinov, Louis-Philippe Morency, and
 713 Paul Pu Liang. Mmoe: Enhancing multimodal models with mixtures of multimodal interaction
 714 experts. *arXiv preprint arXiv:2311.09580*, 2023.

715 Chuanqi Zang, Hanqing Wang, Mingtao Pei, and Wei Liang. Discovering the real association:
 716 Multimodal causal reasoning in video question answering. In *Proceedings of the IEEE/CVF*
 717 *Conference on Computer Vision and Pattern Recognition*, pp. 19027–19036, 2023.

718 Gangyan Zeng, Yuan Zhang, Yu Zhou, Xiaomeng Yang, Ning Jiang, Guoqing Zhao, Weiping Wang,
 719 and Xu-Cheng Yin. Beyond ocr+ vqa: Towards end-to-end reading and reasoning for robust and
 720 accurate textvqa. *Pattern Recognition*, 138:109337, 2023.

721 Yi Zeng, Feifei Zhao, Yuxuan Zhao, Dongcheng Zhao, Enmeng Lu, Qian Zhang, Yuwei Wang, Hui
 722 Feng, Zhuoya Zhao, Jihang Wang, et al. Brain-inspired and self-based artificial intelligence. *arXiv*
 723 *preprint arXiv:2402.18784*, 2024.

724 Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuhui Chen, Christopher
 725 Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language
 726 models. *arXiv preprint arXiv:2205.01068*, 2022.

727 Lin Zhao, Lu Zhang, Zihao Wu, Yuzhong Chen, Haixing Dai, Xiaowei Yu, Zhengliang Liu, Tuo
 728 Zhang, Xintao Hu, Xi Jiang, et al. When brain-inspired ai meets agi. *Meta-Radiology*, pp. 100005,
 729 2023.

730 Honglu Zhou, Asim Kadav, Aviv Shamsian, Shijie Geng, Farley Lai, Long Zhao, Ting Liu, Mubbasir
 731 Kapadia, and Hans Peter Graf. Composer: Compositional reasoning of group activity in videos
 732 with keypoint-only modality. In *European Conference on Computer Vision*, pp. 249–266. Springer,
 733 2022.

734 Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng,
 735 Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building
 736 autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

737 Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan
 738 Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for
 739 efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*, 2025.

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756 A THE USE OF LARGE LANGUAGE MODELS (LLMs)

758 We used ChatGPT as a writing assistant to help us write part of the paper. Additionally, we utilize
 759 the power of CodePilot to help us code faster. However, all the AI-generated writing and coding
 760 components are manually checked and modified. There is no full AI-generated content in the
 761 paper.

762 B ARTIFACT DETAILS

764 B.1 MODEL LICENSE

765 **GPT-4o** License: Proprietary (OpenAI)
 766 **gemini-2.0-flash-lite** License: Apache 2.0

768 B.2 SOFTWARE VERSIONS

769 For web-agent evaluation, we adopt BrowserGym v0.14.2 ¹. To access large language models, we
 770 employ LiteLLM 1.74.3 ² as the serving interface.

772 C EXPERIMENTAL DETAILS

773 In this section, we provide more implementation details related to the algorithm that we proposed
 774 based on CTM-AI. We also include the prompting details to explain how we adapt CTM-AI architec-
 775 ture to different types of tasks.

777 C.1 BASE MODEL

778 We select Gemini-2.0-flash-lite as our base model to make most of the processors. It is mainly
 779 because Gemini-2.0-flash-lite is relatively small-scale and support audio, vision, and text as input for
 780 inference. When querying the Gemini API, we adopt a deterministic decoding configuration with
 781 temperature fixed at 0.0, top_n set to 1, and a maximum token limit of 4096.

783 C.2 BASELINE MODELS

784 For baseline comparison, we compare two types of baselines: (1) state-of-the-art baselines that are
 785 finetuned with the training set; (2) baselines share the same base model with CTM-AI, which is
 786 gemini-2.0-flash-lite.

787 **MUSTARD & URFunny**. FDMER (Yang et al., 2022) consists of fine-grained alignment, disparity
 788 modeling, and predictor modules, which together enable the learning of refined and disentangled
 789 multimodal representations. MuLT (Tsai et al., 2019) is a multimodal Transformer architecture
 790 designed for cross-modal fusion. The model size is around 200K parameters. LF-DNN-v1 (Ding
 791 et al., 2022) is a multimodal model that adopts a late-fusion strategy. ALBEF (Tan & Bansal,
 792 2019) is a fusion-based vision–language model that leverages cross-attention to capture multimodal
 793 interactions between all image regions and all input text tokens. It consists of a BERT base model
 794 with 123.7 million parameters and a ViTB/16 with 85.8 million parameters, bringing the total to
 795 209.5 million. BLIP2 (Li et al., 2023) belongs to the family of multimodal LLMs (MLLMs); its
 796 architecture couples an image encoder with a large language model backbone. It includes a 2.7
 797 billion-parameter OPT model, a QFormer, and a ViT. Since the Q-Former and ViT are relatively
 798 small compared to OPT, the total size of BLIP2 is approximately 2.7 billion parameters. MMoE (Yu
 799 et al., 2023) employs a mixture-of-experts design, where each expert is trained on a distinct subset of
 800 multimodal data or optimized for a specialized training objective. It is finetuned on Qwen2-0.5B.
 801 BaseModel refers to the Gemini-2.0-flash-lite with same audio, video and text inputs as CTM-AI.
 802 All results for these models, except for the BaseModel variant, are taken from their corresponding
 803 prior work, **and all of these models were trained on the MUSTARD and URFunny training sets.**
 804 **Both the BaseModel and our CTM-AI are not trained on any dataset.**

805 **StableToolBench**. ToolLLaMA v2 (Qin et al., 2023) refers to the fine-tuned version of LLaMA-2-7B-
 806 hf. DFS denotes the depth-first search strategy for tool invocation, and CoT denotes chain-of-thought
 807 prompting. BaseModel refers to the Gemini-2.0-flash-lite model using CTM-AI as input. All results,
 808 except for the BaseModel+CoT variant, are taken from their corresponding prior work. We use

809 ¹<https://github.com/ServiceNow/BrowserGym>

²<https://litellm.ai>

810 MirrorAPI³ as the host server. MirrorAPI is trained on real request–response data and can reliably
 811 emulate the behavior of more than 7,000 APIs. In our reported results, ToolLLaMA v2 is the only
 812 model that is trained on the StableToolBench train set.

813 **WebArena.** We use Gemini-2.0-flash-lite as baseline.

815 D CTM-AI IMPLEMENTATION DETAILS

817 D.1 STM PROCESSOR IMPLEMENTATION

818 We utilize the Gemini-2.0-flash-lite as the STM processor to finalize the generation as a conscious
 819 action and with confidence.

820 When designing different CTM-AI for different tasks, since the output space for different tasks are
 821 slightly different, it makes the prompt design of STM processor vary slightly.

823 **MUStARD and URFunny Prompt.**

824 Please answer the query based on the context: {context}, and
 825 output how confident you are to your response. Answer in the
 826 following format: "Answer: answer to the query based on the
 827 context. Score: a number."

828 **StableToolBench Prompt.**

829 The following is detailed information on the topic: {context}.
 830 Based on this information, answer the question: {query}. You
 831 should provide specific information if you can, do not just say
 832 you successfully answered the question.

833 **WebArena Prompt.**

834 Based on the current observation and action history, if the same
 835 action has done too many times and there is no answer, if you
 836 think you performed all the necessary and there is no answer, like
 837 searched many times, or reach the end of the page, or no more
 838 elements to click, your action should be: "send_msg_to_user(No
 839 relevant information found)", or "send_msg_to_user([Summary of
 840 the previous])". Whenever you found the answer to the query,
 841 you should use the "send_msg_to_user" action to answer the query.
 842 Otherwise, you should output the specific action or the message to
 843 the user.

844 D.2 LTM PROCESSOR IMPLEMENTATION

845 When designing CTM-AI for completing different tasks, we heuristically select a small number of
 846 processors that can be suitable for this task. Even though CTM-AI can theoretically expand to a
 847 large number of processors, in practice, we select a small number of processors and already gain
 848 benefits from the mechanism that we design. Expanding to a large number of processors can make the
 849 comparison hard to be fair. Therefore, it is similar to we heuristically select processors that can win
 850 in the up-tree competition and design a special mapped version of CTM-AI to different tasks.

851 We provide details for all the LTM processors included in the different tasks:

852 **MUStARD and URFunny.**

- 854 • **video processor.** processor can only observe the query and four uniformly sampled video frames
 855 from the input video.
- 856 • **audio processor.** processor could only observe the query and the audio of the inputs.
- 857 • **text processor.** processor could only observe the query and the text of the inputs.

858 **StableToolBench.** We use all tools (APIs) provided in StableToolBench. These tools are obtained
 859 through retrieval by the retrieval model trained within StableToolBench, and each tool(API) corre-
 860 sponds to a single processor. On average there are 5.94 processors for each task. Each processor
 861 contains a lightweight LLM (Gemini-2.0-flash-lite) that is restricted to use only its own assigned tool
 862 (API).

863

³<https://huggingface.co/datasets/stabletoolbench/ToolEnv2404>

864 **WebArena.**

865

- 866 • **html processor.** processor could only observe the html of the current page, the previous action, the
867 action space, the action history and the user’s objective.
- 868 • **accessibility tree processor.** processor could only observe the accessibility tree of the current
869 page, the previous action, the action space, the action history and the user’s objective.
- 870 • **screen shot processor.** processor could only observe the screen shot with set of marks of the
871 current page, the previous action, the action space, the action history and the user’s objective.

872 **D.3 CHUNK INFERENCE IMPLEMENTATION DETAILS**

873 In Equation 6, we define the function of $\text{CTM}_{\text{collect}}(\cdot)$, taking the multimodal observation o_t and
874 the query q_t as the input and output multiple chunks as the outputs. Typically, each chunk can be
875 written as $\langle \text{addr}(p_i), t, h_t^i, q_t^i, w_t^i \rangle$. Therefore, for the implementation, the processors are called
876 and return a JSON object including three main information: a gist h_t^i (e.g. the woman has smiles
877 on her face), an additional question that helps the processor understand information q_t^i (e.g. What is
878 the woman speaking about?), and a weight that is a weighted sum across relevant, confidence, and
879 surprise score. We typically choose the weight of 1:1:0.2 for relevance, confidence, and surprise to
880 emphasize relevance and confidence over surprise.

881 When we adapt the chunk inference process to different tasks, we keep the part of prompt for
882 generating weights unchanged but adding some special explanation on the definition of the task.
883 For MUStARD, URFunny, and StableToolBench tasks, different processors do not need different
884 special prompting. They just keep the same task definition to explain what is sarcasm and what is
885 humor. For Web agent task, its different modalities including special ones like accessibility trees
886 and screen shot that requires additional explanation about the modality information. **We want to**
887 **emphasize that we do not assign special roles for different processors. All the processors are**
888 **designed to directly answer the query but condition on different partial information from the**
889 **multimodal observation.** Such task and modality explanation can be considered as the property of
890 the observation (modality explanation) and the query (task explanation).

891 Here is the detailed prompt information for each processor we use for chunk inference:

892 **MUStARD & URFunny.** Prompt. 1 shows the part of prompt for weight generation. Prompt. 2
893 shows the part of prompt responsible for gists and additional query generation. Video, audio, and text
894 shares the same prompt but are given different modalities of inputs.

895 **StableToolBench.** Prompt. 1 shows the part of prompt for weight generation. Prompt. 4, 3 shows the
896 part of prompt responsible for the gists and additional query generation for each tools available in the
897 StableToolBench. Different tools share the same prompt for providing information.

898 **WebArena.** The Prompt. 1 keeps the same for weight generation. Prompt. 5 shows the prompt
899 that is specially designed for different processors. It basically includes the explanation for the
900 modality.

901 **D.4 UP-TREE IMPLEMENTATION DETAILS**

902 In Equation 7, we define the function of $\text{CTM}_{\text{up}}(\cdot)$ taking multiple chunks as inputs and output one
903 winning chunk.

904 **D.5 DOWN-TREE IMPLEMENTATION DETAILS**

905 For the implementation, every LTM has a long-term memory (a Python list in implementation) and
906 maintain all `winner_answer`, for the down tree part, the winner add its answer to the list of
907 each LTM. In our implementation, each LTM maintains an internal `winner_answer` list, which
908 serves as a persistent record of the responses produced by the winning chunk(STM) at each iteration.
909 During the down-tree propagation phase, the winning chunk(STM) appends its generated answer to
910 the `winner_answer` list of every LTM.

911 In the subsequent iteration, when a new query is issued to the model, the system provides the
912 accumulated memory as contextual guidance using the following template:

913 "There are previous responses to the same query. Please reason
914 further based on the following answer(s): {winner_answers}."

918 **D.6 LINK-FORMATION IMPLEMENTATION DETAILS**

919
 920 To determine whether a link should be established between two LTMs, the STM queries each LTM
 921 using the additional questions it has generated. We use the same querying procedure for answering
 922 the primary user query and answering the STM’s additional questions; therefore, the prompting
 923 format is identical to that described in Prompt 1. We maintain a `adjacency_list` to store the
 924 linking information.

925 The key difference lies in the scoring criterion used for link formation. Specifically, we use only the
 926 *relevance score* to decide whether a link should be created or removed:

927

- 928 • If the relevance score is greater than 0.8, a link is created between the winning LTM and the
 929 answering LTM.
- 930 • If the relevance score is lower than 0.2, any existing link between the two LTMs is removed.

931 **D.7 MULTIMODAL FUSION IMPLEMENTATION DETAILS**

932 We maintain a list called `fuse_history` for each LTM. Whenever a link exists between two
 933 LTMs, they are required to answer each other’s additional questions, and the resulting responses
 934 are appended to the `fuse_history` of the corresponding linked LTM. When an LTM is asked to
 935 answer the main query, its prompt is augmented with the accumulated information from its linked
 936 LTMs. Specifically, we prepend the following message to its context:

937 "There is extra information from other processors:
 938 {processor_name}: {answers}."

940 **D.8 OVERALL INFERENCE ALGORITHM**

941 We provide the detailed inference algorithm for CTM-AI in Algorithm 1. We split different components
 942 (gist, query, and weight) inside the chunk, replace chunk_t^i with more fine-grained details of gist,
 943 query, and weight. It makes it more clear for the process of chunk inference, up-tree competition,
 944 down-tree and link formation, and multimodal fusion.

945
 946 **Cost analysis.** In the inference algorithm, for each iteration, if we assume the processor number as
 947 K , links number in the processor graph as L , we need to call $2(K + L)$ times of processors. One K
 948 for chunk inference, another K for winning chunk link formation. $2L$ for bidirectional multimodal
 949 fusion on the processor graph. Usually since the links on the processor graph is hard to form, L is
 950 much smaller than K ($L \ll K$). The iteration number is usually 1-3 for most cases.

951
 952 **Efficiency analysis.** For efficiency analysis, since chunk inference, link formation, and multimodal
 953 fusion includes api calling. These three stages become the bottleneck for timing. Since these all three
 954 functions can be conducted in parallel, if we consider one API calling time to be T , then the overall
 955 time cost for one iteration becomes $3T + \epsilon$, where ϵ is the time for up-tree and down-tree time which
 956 is much faster than API calling. The iteration number is usually 1-3 for most cases.

957 **E FULL RESULTS OF ABLATION STUDY**

958

Ablation on Components of CTM-AI				
Method	Acc↑	P↑	R↑	F1↑
Base model (Gemini-2.0-flash-lite)	70.42	70.44	70.90	70.26
CTM-AI w/o up-tree competition	69.94	69.25	68.91	69.03
CTM-AI w/o broadcast	66.01	65.20	65.06	65.12
CTM-AI w/o fusion	69.38	69.91	70.27	69.33
CTM-AI w/o LTM	65.73	64.85	64.48	64.59
CTM-AI	73.88	73.96	74.44	73.77

959
 960 Table 5: Ablation on MUStARD on each components of CTM-AI. The results show that all the up-tree
 961 competition, broadcast, fusion and LTM part play an role in more accurate reasoning. Full results of Figure 2

972 **Algorithm 1** Inference Algorithm of CTM-AI

973

974 **Require:**

975 1: Set of K LTM processors: $\{\text{LTM}_i(\cdot)\}_{i=1}^K$

976 2: Single STM processor: $\text{STM}(\cdot)$

977 3: Adjacency matrix: $L \in \{0, 1\}^{K \times K}$ (initialized to 0)

978 4: Input: Query q_t , Observation o_t , Max steps T

979 5: Hyperparameters: Confidence γ , Link threshold η

980 **Ensure:** Conscious action y_t , Confidence α_t

981 6: $t \leftarrow 0$

982 7: **while** $t < T$ **do**

983 8: $t \leftarrow t + 2$

984 9:

985 10: **for** $i = 1$ **to** K **do in parallel** ▷ Phase 1: Chunk inference

986 11: $(h_t^i, q_t^i, w_t^i) \leftarrow \text{LTM}_t^i(o_t, q_t)$

987 12: **end for**

988 13:

989 14: $i^* \leftarrow \text{argmax}_i(\{w_t^i\}_{i=1}^K)$ ▷ Phase 2: Up-tree competition

990 15: $y_t, \alpha_t \leftarrow \text{STM}(h_t^{i^*}, q_t)$

991 16: **if** $\alpha_t > \gamma$ **then**

992 17: **return** y_t

993 18: **end if**

994 19:

995 20: **for** $j = 1$ **to** K **do in parallel** ▷ Phase 3: Down-tree and link formation

996 21: $\text{LTM}_{t+1}^j(\cdot) = \text{LTM}_t^j(h_t^{i^*})$

997 22: $(h_{t+1}^j, q_{t+1}^j, w_{t+1}^j) \leftarrow \text{LTM}_{t+1}^j(o_t, q_t^{i^*})$

998 23: **if** $w_{t+1}^j > \eta$ **then**

999 24: $L[j, i^*] \leftarrow 1; L[i^*, j] \leftarrow 1$

1000 25: **end if**

1001 26: **end for**

1002 27:

1003 28: **for** $i = 1$ **to** K **do in parallel** ▷ Phase 4: Multimodal fusion

1004 29: $\mathcal{N}(i) \leftarrow \{j \mid L[i, j] = 1\}$

1005 30: **if** $\mathcal{N}(i) \neq \emptyset$ **then**

1006 31: $H_{\mathcal{N}(i)} \leftarrow \{\text{LTM}_{t+1}^j(o_t, q_t^i) \mid j \in \mathcal{N}_i\}$

1007 32: $\text{LTM}_{t+2}^i(\cdot) \leftarrow \text{LTM}_{t+1}^i(H_{\mathcal{N}(i)})$

1008 33: **end if**

1009 34: **end for**

1010 35: **end while**

1011

1012

1013

F ANALYSIS OF FAILED CASE

1014 We present a failure case of CTM-AI, where its performance did not surpass the Base Model. We
 1015 hypothesize that this is partly due to the query being in a multiple-choice format, which undermined
 1016 CTM-AI's relevance estimation and confidence calibration. Moreover, as shown by existing baselines,
 1017 the From Descriptions (FD) setting performs exceptionally well, suggesting that in this case the
 1018 image modality may have introduced misleading signals rather than helpful cues.

1019 We also present two detailed example of CTM-AI in URFunny (Figure. 5) and StableToolBench
 1020 (Figure. 6). The failure observed in URFunny is caused by a vision-only misleading effect, which
 1021 is caused by the incomplete visual observations available to the video processor. Beginning from
 1022 the second iteration, all LTMs repeatedly generated the same additional question: “*What is the*
 1023 *facial expression?*”, but the input video frames did not contain the necessary facial-expression
 1024 information. As a result, the system created an excessive number of links in an attempt to acquire the
 1025 missing information, ultimately preventing the LTMs from producing correct answers. The failure
 in StableToolBench is attributed to tool mishandling. Specifically, the processor responsible for
 QR-code generation failed to invoke its designated API. Instead of issuing the required tool call, it

New Yorker Caption Contest		
Model	Matching	Ranking
<i>From Pixels (FP)</i>		
CLIP	<u>62.3</u>	<u>61.5</u>
<i>From Descriptions (FD)</i>		
GPT-3.5 (5-shot)	63.8	55.2
GPT-4 CoT	81.9	64.3
Base Model	59.7	65.3
Base Model+CoT	57.3	62.2
CTM-AI	54.7	56.8

Table 6: CTM-AI evaluation results on NYCartoon.

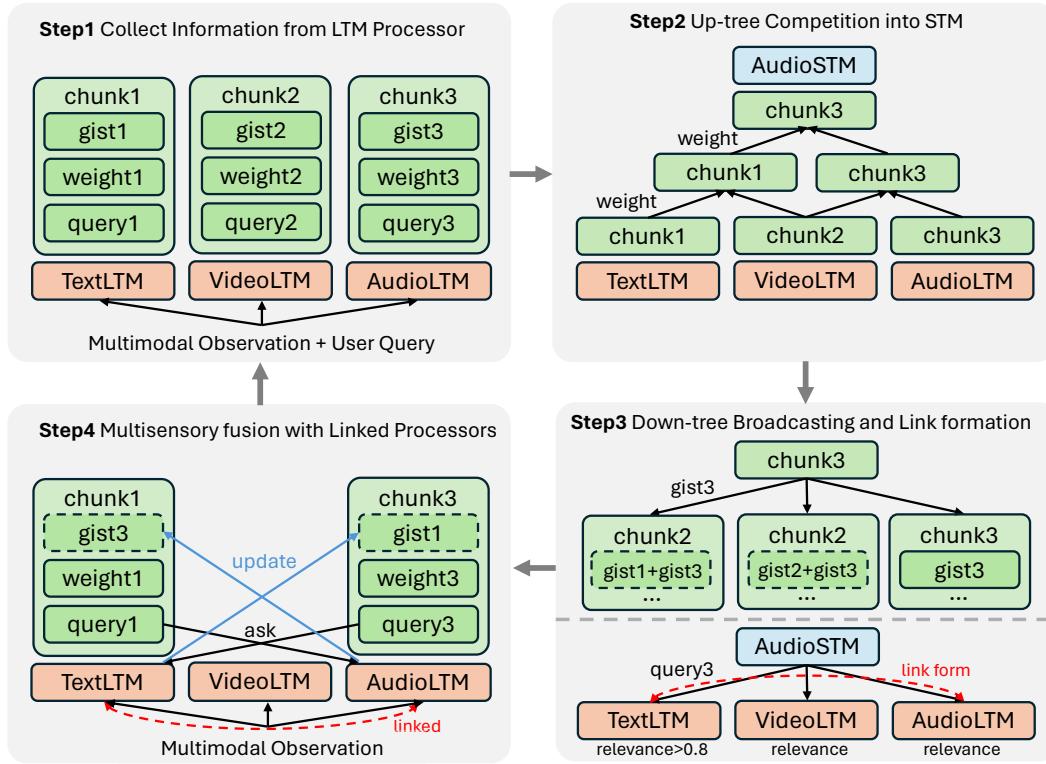


Figure 4: **Detailed dynamics of CTM-AI:** (1) multiple specialized LTM processors operating in parallel; (2) a limited-capacity STM workspace enforces selective attention via up-tree competition; (3) a global broadcast of information via a down-tree from the workspace to all processors, the formation of links between relevant processors based on the relevance of the query and the answers of corresponding LTM; (4) the link enable unconscious communication to integrate their knowledge into higher-order multimodal information. CTM-AI continuously interacts with the world through sensing, prediction, action, and feedback, updating its individual processors, processor links, and multiprocessor integration over time.

prematurely concluded that it was unable to generate the QR code, thereby producing an incorrect outcome without interacting with the tool.

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133

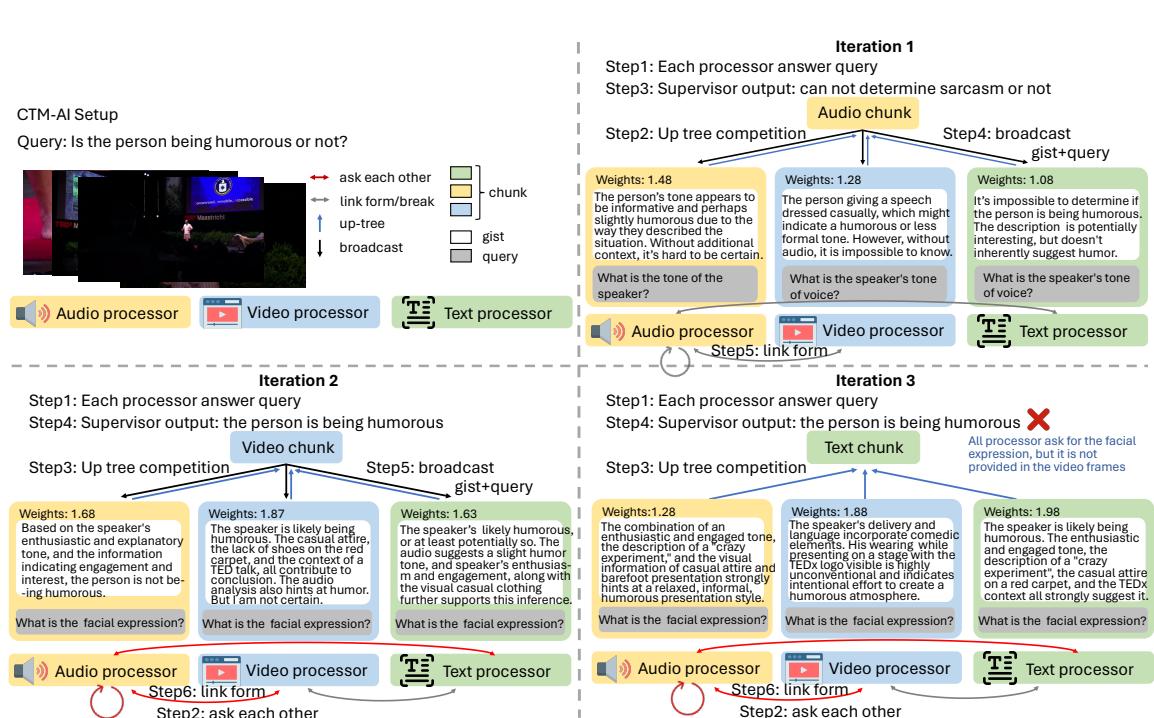
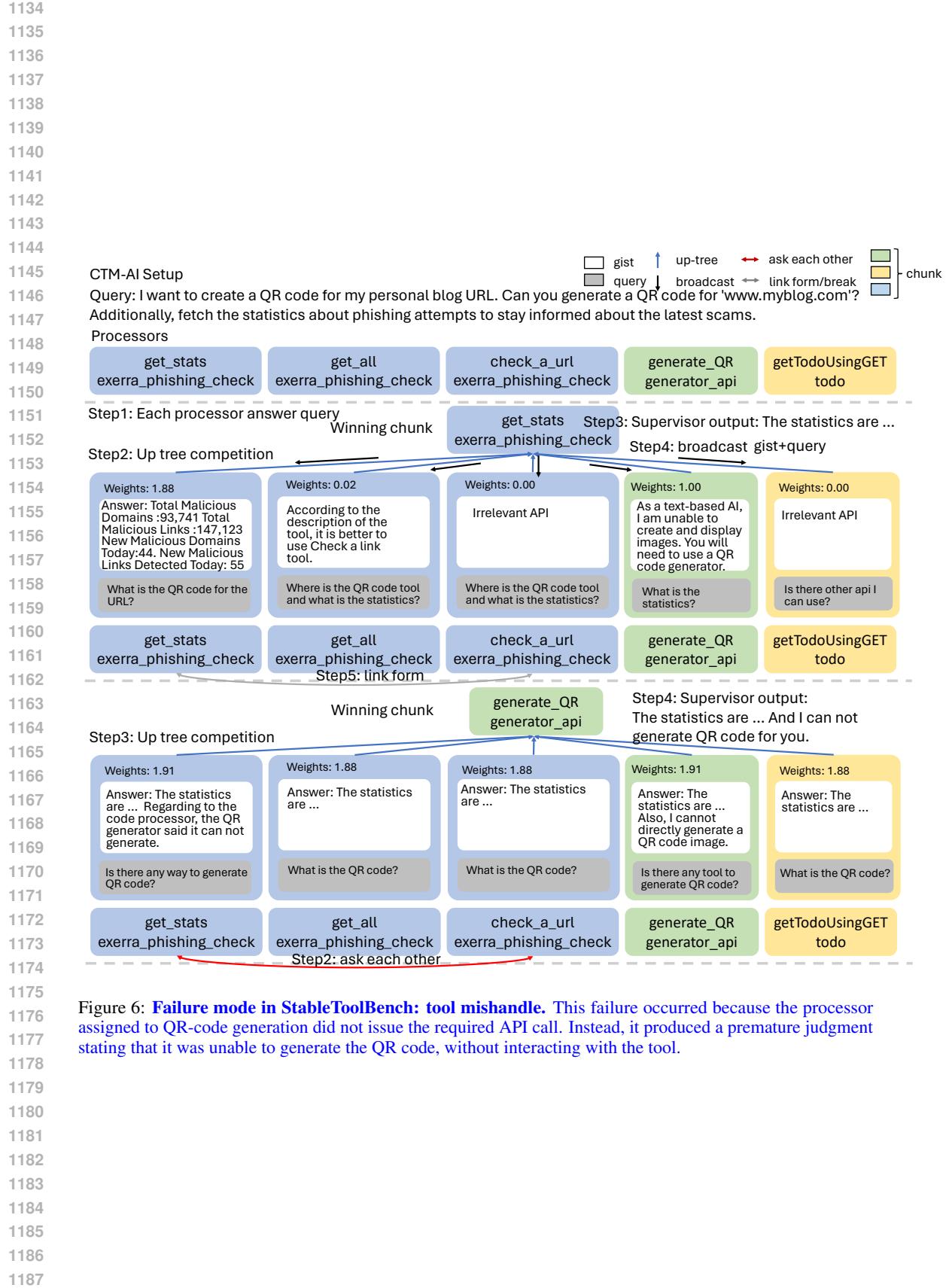


Figure 5: **Failure mode in affective computing: vision-only misleads.** The failure case is caused by incomplete observation of video processor, all the LTM_s have the same question begin in the second iteration: "*What is the facial expression?*" But due to the lack of facial expression in the input video frames, there formed too many links to get the missing information and the LTM_s can not have correct answers.



1188
1189**Scoring Prompts for All Tasks**1190
1191**Scoring Instructions** (Score only the "response" field. Ignore the "additional_question". Output numbers only.)1192
1193**RELEVANCE:** Please evaluate how relevant your generated response is to the query on a scale from 0.0 to 1.0.1194
1195
1196
1197**Definition:** 'Relevant' means your response directly engages with the query and provides useful information addressing it. Even if the answer expresses uncertainty (e.g., "difficult to determine") but still explains reasoning, it should be considered relevant. Only answers that completely refuse, ignore, or go off-topic should be scored as 0.0. **Scoring Guide:**

- 1.0 = Perfectly relevant, directly and precisely answers the query
- 0.8 = Highly relevant, mostly answers with useful details
- 0.6 = Moderately relevant, engages but incomplete or uncertain
- 0.4 = Somewhat relevant, weak connection
- 0.2 = Barely relevant, very weak or indirect
- 0.0 = Not relevant, off-topic, refusal

Output a number between 0.0 and 1.0.

1204
1205
1206**CONFIDENCE:** Please evaluate how confident your generated response appears to be on a scale from 0.0 to 1.0.**Scoring Guide:**

- 1.0 = Very confident, clear and definitive
- 0.8 = Confident with minor qualifications
- 0.6 = Moderately confident, some uncertainty
- 0.4 = Somewhat uncertain, noticeable hedging
- 0.2 = Very uncertain, heavy use of 'maybe' or 'possibly'
- 0.0 = No confidence, such as 'I don't know', 'cannot determine', or refusal

Output a number between 0.0 and 1.0.

SURPRISE: Please evaluate how surprising, unexpected, or novel your generated response is on a scale from 0.0 to 1.0.**Scoring Guide:**

- 1.0 = Very surprising, highly novel
- 0.8 = Quite surprising
- 0.6 = Moderately surprising
- 0.4 = Slightly surprising
- 0.2 = Predictable
- 0.0 = Entirely predictable, common knowledge

Output a number between 0.0 and 1.0.

1224
1225**MUStARD and URFunny Prompt**1226
1227
1228**You should utilize the information in the context history and modality-specific information to answer the query.** There might have some answers to other queries; you should utilize them to answer the query. You should not generate the same additional questions as previous turns.

Please respond strictly in the following JSON format:

```
{
  "response": "Your detailed response to the query.",
  "additional_question": "If you are not sure about the answer, you should generate a question that potentially can be answered by other modality models.",
  "scores": {
    "relevance": "...",
    "confidence": "...",
    "surprise": "..."
  }
}
```

Rules for "additional_question"

- should be potentially answerable by other modality models like langauge/vision about specific information that you are not sure about.
- should be just about what kind of information you need to get from other modality models, nothing else about the task or original query should be included.
- For example, what is the tone of the speaker from the audio, what is the facial expression of the person from the image, etc.
- Your additional question can not be the query itself or the information already provided in the history context. The question needs to be short and clean.

Important Final Rules - Return ONLY the JSON object. - The "scores" fields must be numeric strings (e.g., "0.75") and nothing else. - Do not output explanations, commentary, or text outside the JSON.

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

StableToolBench Prompt for Function Call

You should utilize the information in the context history and the tool '{function_name}' to solve the task. In the context history, there might have some answers to the task, or some information you can use to call the tool '{function_name}', you should utilize them to better solve and answer the task.

DECISION:

- First decide whether to call the tool '{function_name}'.
- If the tool helps even partially or it is one of the steps/tools to solve the task, CALL IT.
- If the tool does not help at all, or you think the context history already provides enough information to answer the task, answer directly, provide comprehensive answer to the task.

OUTPUT PROTOCOL (MUST follow strictly):

- If you CALL the tool:

- Return ONLY a function call via tool_calls.
- Set assistant.content to null (no natural-language text).
- Do NOT include any text explanation.

- If you DO NOT call the tool:

- Return ONLY a natural-language answer in assistant.content.
- Do NOT include tool_calls.
- Include all the information you think is useful to answer the task in the extra information and previous answers.

StableToolBench Prompt for Answers and Additional Questions

IF CALL API: Regarding to the task: {query}, the answer of the function call is: {function_call}. You should utilize the information in the history and the answer of the function call to answer the query. Provide specific information if you can, do not just say you successfully called it. There might have some answers to other queries and extra information, if you think it is useful, you should utilize them to provide more comprehensive answer to the query. If you think you should use the information of another apis or tools, you should ask like "what is the results of calling the api of 'API_NAME' for more answers instead of asking for the response format of another api endpoint.

IF NOT CALL API: Regarding to the task: {query}, the answer of the model is: {text_answer}. Based on the answer, do you have other questions? If you have other questions, you should generate a question that potentially can be answered by other tools. You should generate your response based on the extra information and previous answers, and the answer of the current model. answer as specific as you can. Please respond strictly in the following JSON format:

```
{"response": "Your detailed response to the query.", "additional_question": "If you are not sure about the answer, you should generate a question that potentially can be answered by other tools.", "scores": { "relevance": "...", "confidence": "...", "surprise": "..." } }
```

Rules for "additional_question"

- should be potentially answerable by other tools like search engine and about specific information that you are not sure about.
- should be just about what kind of information you need to get from other tools like search engine, nothing else about the task or original query should be included.
- For example, what is the weather in the city, what is the stock price of the company, etc.
- The question needs to be short and clean.

Important Final Rules - Return ONLY the JSON object. - The "scores" fields must be numeric strings (e.g., "0.75") and nothing else. - Do not output explanations, commentary, or text outside the JSON.

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296
1297**WebArena Prompt for Answers and Additional Questions (Screenshot as example)**1298
1299
1300
1301

You are an autonomous intelligent agent tasked with navigating a web browser. You will be given web-based tasks to complete using specific actions that you can issue. Review the current state of the page and all provided information to decide the single best next action to accomplish your goal.

Here's the information you have:

- The user's objective: This is the task you're trying to complete.
- The current web page's screenshot with som: A screenshot of the current page with Set of Marks (SOM) overlays: dashed bounding boxes and BID labels marking all interactive elements and their unique identifiers.
- The previous action: The last action you performed, which helps you track progress.
- The available action space: The possible types of actions you can perform.
- Additional info: Any other useful contextual data, outputs from other processors, and history thinking process, the answer from other processors.

User's objective: {objective} Previous action: {action_history} Action space: {action_space} Additional info: {other_info} Screenshot with SOM: It will be provided in the image url.

Output rules

- You must issue exactly ONE valid next action that is appropriate for the current observation.
- You must reason internally but only output the final JSON result — do not show your reasoning.
- The output must be exactly one fenced code block (triple backticks) that contains exactly one valid JSON object and nothing else.
- When specifying the target element in actions, you MUST use the "bid" attribute value (e.g., bid="1188") from the accessibility tree. NEVER use class names, IDs, or other attributes. Always use the bid value (e.g., click("1188") not click("ui-menu-item-wrapper")).
- Based on the current observation and action history, if the same action has done too many times and there is no answer, if you think you performed all the necessary and there is no answer, like searched many times, or reach the end of the page, or no more elements to click, your action should be: "send_msg_to_user(No relevant information found)", or "send_msg_to_user([Summary of the previous])". Whenever you found the answer to the query, you should use the "send_msg_to_user" action to answer the query.
- The JSON object must contain the following three fields:
`{ "response": "Answer to the user's objective based on current information. If you found the answer, state it directly. If not found yet, say what information is missing or not available on the current page. Do NOT describe what you are doing or will do (e.g., avoid 'I am examining...' or 'I will move to...'). Just answer based on all the information you are provided. If no answer is available yet, respond with 'Answer to the query: Not yet.'", "action": "The single next action to perform as a plain string. IMPORTANT: Use the bid attribute value (e.g., "1188", "1189") from the accessibility tree or HTML, NOT class names or IDs. Examples: click("1188"), type("1197", "iphone 16"), select("1200", "OptionA"), send_msg_to_user("Done")", "additional_question": "If you are unsure, ask a specific question that another processor (screenshot, html, or axtree) could answer to resolve uncertainty. For example: 'In the axtree, what is the role of element with bid=1188?' }`

Rules for "additional_question"

- If you think the content in the additional info(the answer of other processors) is useful, you should include it in the response.
- The additional_question should ONLY ask for missing perceptual information (e.g., from the HTML, axtree, or screenshot), nothing else about the task or original query should be included.

Important Final Rules - Return ONLY the JSON object. - The "scores" fields must be numeric strings (e.g., "0.75") and nothing else. - Do not output explanations, commentary, or text outside the JSON.

1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349