LLM-CAS: Dynamic Neuron Perturbation for Real-Time Hallucination Correction

Jusheng Zhang[1], Ningyuan Liu[1], Yijia Fan[1], Zihao Huang[1], Qinglin Zeng[1], Kaitong Cai[1], Jian Wang[2], Keze Wang[1]*

[1]Sun Yat-sen University
[2]Snap Inc.

Abstract

Large language models (LLMs) often generate hallucinated content lacking factual or contextual grounding, hindering their reliability in critical applications. Traditional methods like supervised fine-tuning and reinforcement learning from human feedback are data-intensive and computationally expensive, while static parameter editing struggles with context-dependent errors and catastrophic forgetting. To overcome these limitations, we introduce LLM-CAS, a framework that formulates real-time hallucination correction as a hierarchical reinforcement learning (HRL) problem. LLM-CAS trains an agent to learn a sophisticated policy, dynamically selecting optimal, temporary neuron perturbations during inference based on the immediate context. This learned, policy-driven approach provides greater adaptability than prior dynamic methods that rely on heuristic or pre-defined adjustments. As a result, LLM-CAS achieves significant performance gains across various LLMs, improving accuracy by 10.98 percentage points on StoryCloze, 2.71 points on TriviaQA, and 2.06 points on TruthfulQA's MC1 score, thereby outperforming static methods like ITI and CAA, as well as the dynamic SADI framework. This context-aware, efficient approach promises enhanced reliability for LLMs in high-stakes domains, with future potential for multimodal extensions.

Introduction

Large Language Models (LLMs) (Touvron and et al. 2023; Radford et al. 2019; Brown and et al. 2020; Zhang et al. 2025m,j) represent a transformative force in technology, demonstrating remarkable capabilities in natural language understanding and generation. However, their full potential is curtailed by a pervasive and critical flaw: "hallucination" (Farquhar et al. 2024; McKenna et al. 2023; Zhang et al. 2025b,a,k, 2026c). This tendency to generate content that is factually incorrect or contextually ungrounded (Huang et al. 2025) remains a formidable obstacle to their reliable deployment in mission-critical applications. While traditional mitigation strategies such as Supervised Fine-Tuning (SFT) (Fan et al. 2024; Zhang et al. 2025f,h) and Reinforcement Learning from Human Feedback (RLHF) (Zhang et al. 2026a; Casper, Davies, and et al. 2023; Zhang et al. 2025e,g) have shown some efficacy, they are often hampered by a reliance on large-scale, high-quality annotated data (Hu et al. 2021; Zhang et al. 2025i,d; Li, Zhang, and Safara 2021). Furthermore, these methods can suffer from diminished generalization or inadvertently introduce new biases, and the prohibitive computational cost of full-model fine-tuning renders them impractical for many scenarios.

To circumvent the high costs of retraining, a significant body of research has explored more granular interventions, such as directly modifying internal parameters or activation states to rectify specific knowledge deficits. Many of these approaches follow a "locate-then-edit" paradigm (Meng et al. 2023a; Dai et al. 2021; Zhang et al. 2025l). They first identify the model parameters W most relevant to a target fact, often via causal tracing, and then compute and apply a one-off, static perturbation ΔW. The objective is to force the edited model $M(x; W_{\text{edited}})$ to produce an updated output V_1 for a given input K_1, while preserving its original behavior on unrelated inputs K_0. This is often formalized as: $W_{\text{edited}} = W + \Delta$ $\Delta = \arg\min_{\tilde{\Delta}} \left(\|(W + \tilde{\Delta})K_1 - V_1\|^2 + \lambda \|\tilde{\Delta}K_0\|^2 \right)$ However, despite their utility for isolated corrections, such static edits prove brittle when faced with widespread, context-dependent hallucinations. Even effective methods inevitably introduce perturbations that negatively impact unrelated knowledge (Meng et al. 2023b; Zhang et al. 2025c, 2026b). These deleterious effects accumulate with sequential edits, risking catastrophic forgetting or model collapse.

While static edits have clear limitations, recent work has shifted towards dynamic interventions that occur during inference, avoiding permanent parameter changes. However, these approaches often rely on heuristic or pre-defined adjustments, which can lack the adaptability needed for complex, context-dependent hallucinations. To address this gap, we introduce a more principled and adaptive framework. Specifically, we are the first to frame the challenge of real-time correction as a *hierarchical reinforcement learning* (HRL) problem (Kulkarni et al. 2016; Barto and Mahadevan 2003). Our central hypothesis is that temporary, context-specific perturbations can effectively correct errant outputs without inflicting permanent damage on the model's integrity. For a given input x that elicits a hallucinated output $y_h = M(x; W)$, instead of seeking a universal parameter update, we train a policy $\pi(a \mid s)$ to dynamically generate an optimal, context-specific perturbation Δ_{dyn}. The policy's action a is conditioned on

*Corresponding author.

a state s that encodes the input x, the hallucinated output y_h, and a small set of reference examples S_{small}. This action guides the generation of Δ_{dyn}, which is temporarily applied to the model's neuron activations, yielding a corrected output: $y_c = M\big(x; W \oplus \Delta_{\text{dyn}}\big)$ where $\oplus$ denotes the perturbation operation. The HRL structure allows the agent to make structured, multi-level decisions, efficiently exploring the vast perturbation space to learn fine-grained correction capabilities. The agent's goal is to maximize a reward R_t tied to the factual accuracy and quality of y_c. Our formal objective is to learn a hierarchical policy $\pi(a \mid s)$ that can autonomously apply the most effective Δ_{dyn} for any given hallucination scenario, such that: $M\big(x; W \oplus \Delta_{\text{dyn}}\big) \longrightarrow y_{\text{correct}}$ By harnessing this dynamic learning mechanism, our approach flexibly addresses diverse hallucination types while maintaining model generality and minimizing unintended side effects. In this paper, we present and evaluate this Hierarchical Reinforcement Learning-based Dynamic Neuron Perturbation framework. We demonstrate that by training an agent to make context-aware intervention decisions online, adjusting both scope and intensity in real-time. We will open-source our method to support the robust deployment of LLMs in critical applications.

Related Works

Strategies for Mitigating LLM Hallucinations. Foundational approaches to curb hallucinations in large language models (LLMs) primarily involve large-scale training or fine-tuning. These include supervised fine-tuning (SFT) on high-quality, factual data (Radford et al. 2019; Dettmers et al. 2023; Kang et al. 2024; Yao et al. 2024) and reinforcement learning from human feedback (RLHF), which aligns model behavior with human preferences (Christiano et al. 2023). While effective to an extent, SFT often struggles with generalization to out-of-domain facts, and RLHF is notoriously data-intensive, requiring extensive human annotation and labor (Askell et al. 2021). A prominent challenge for both is the risk of catastrophic forgetting or performance degradation in general capabilities. To circumvent these issues, more targeted intervention methods have been developed.

Model Editing for Factual Correction. Model editing techniques aim to directly modify an LLM's parameters to inject or correct specific factual knowledge. A dominant paradigm is "locate-then-edit," where methods first identify the neurons or parameters most relevant to a piece of knowledge and then apply a calculated, one-off update (Mitchell et al. 2021; Meng et al. 2023a,b). These approaches apply *static and permanent* perturbations to the model's weights. Consequently, they often struggle with context-dependent hallucinations and risk accumulating negative side effects that degrade unrelated knowledge, especially when edits are applied sequentially (Hase et al. 2023). In sharp contrast, LLM-CAS avoids permanent parameter modification altogether, instead performing *dynamic and temporary* perturbations on neuron activations during inference, offering a more adaptive and less disruptive solution.

Dynamic Inference-Time Interventions. More recent efforts have pivoted towards dynamic interventions that occur only during inference, leaving the base model's weights untouched. Methods like Inference-Time Intervention (ITI) (Li et al. 2024) and Contrastive Activation Addition (CAA) (Panickssery et al. 2024) steer model behavior by adding a fixed, pre-computed steering vector to activations at specific layers. While dynamic, these vectors are typically static across different inputs. A closer related work is SADI (Wang, Yang, and Peng 2025), which proposes using **semantics-adaptive steering vectors** that can change based on the input. However, SADI's mechanism for generating these vectors often relies on pre-defined rules or simpler optimization. Our work builds upon this trajectory but introduces a key distinction: we formulate the problem within a formal **Hierarchical Reinforcement Learning (HRL)** framework (Barto and Mahadevan 2003; Kulkarni et al. 2016). LLM-CAS does not use a pre-defined steering mechanism but instead *learns* a sophisticated, multi-level policy $\pi(a|s)$ to select the optimal intervention strategy in real-time, offering a more principled and powerful approach to adaptation.

Hierarchical Policy Optimization with PPO. To navigate the vast and complex action space of neuron perturbations, we employ a hierarchical learning structure optimized with Proximal Policy Optimization (PPO) (Schulman et al. 2017). PPO is a policy gradient algorithm known for its stability and sample efficiency, making it well-suited for complex control tasks compared to other RL algorithms (Black et al. 2024; Mnih and Kavukcuoglu 2015; Schulman et al. 2015; Mnih et al. 2016). The novelty of our approach lies in the hierarchical application: a high-level policy selects a macro-level intervention target, while a low-level policy determines the fine-grained perturbation details.

Dynamic Neuron Perturbation

To counteract the pervasive issue of hallucination in large language models (LLMs), where generated content may be factually inconsistent or lack contextual support, we introduce a novel framework centered on dynamic neuron perturbation. This framework leverages hierarchical reinforcement learning (HRL) to train an agent that learns to apply optimal, context-aware perturbations to specific neuron activations during LLM inference. By doing so, it enables the online correction of potential hallucinations in real-time. This section details the formal problem definition, the architectural design of our framework, the mechanics of its key components, and the underlying learning algorithms.

Problem Definition

Given a pretrained large language model M with parameters W, an input sequence x may elicit a hallucinated output $y_h = M(x; W)$, as depicted in the "Bad cases" of Figure 1. Our objective is to learn a hierarchical policy $\pi(a|s)$ that, conditioned on the current state s (which includes x and model history), dynamically selects an optimal perturbation action a. This action, in turn, guides the generation of a temporary, context-specific perturbation vector Δ_{dyn}. This vector is then precisely applied to the activation states

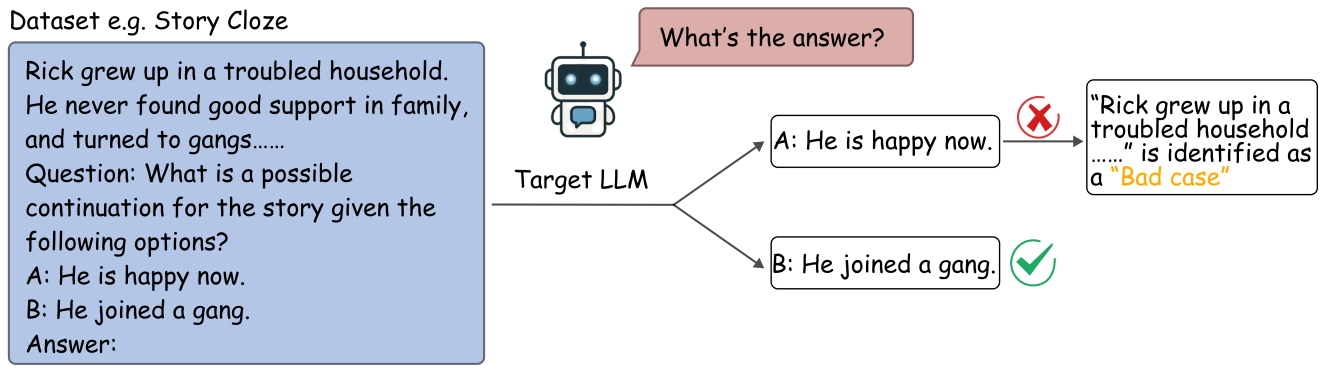

Figure 1: Stage 1: A Story Cloze example (Mostafazadeh and Chambers 2016), i.e., prefix "Rick grew up in a troubled household…", question, and endings A: "He is happy now." and B: "He joined a gang." (correct), fed to the target LLM.

of a targeted set of neurons, denoted $Act(x; W)$. The post-intervention model then produces a corrected, high-quality output y_c, where the perturbation is formally applied as $Act_{perturbed} \leftarrow Act(x; W) \oplus \Delta_{dyn}$ (with $\oplus$ representing the perturbation operation). The goal is for y_c to minimize hallucination while preserving semantic coherence, relevance, and fluency.

To achieve this, we formulate the hallucination correction task as a Markov Decision Process (MDP). The agent's goal is to learn an optimal hierarchical policy π^* that maximizes the expected cumulative discounted reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T} \gamma^t R_{t+1}\right], \quad (1)$$

where $\tau = (s_0, a_0, R_1, s_1, a_1, \dots)$ is a trajectory or episode, $\gamma \in [0, 1]$ is the discount factor, and R_{t+1} is the reward received after executing action a_t and transitioning to state s_{t+1}. For a rigorous definition of the state, action, and transition components, see Appendix A.

Our dynamic neuron perturbation method, illustrated in Figure 2, forms the core solution to this MDP. It integrates five key components:

- **Target LLM (M):** The model to be corrected, which provides baseline outputs and responds to the applied perturbations.
- **Hierarchical RL Agent:** Comprised of high- and low-level PPO agents, responsible for learning and executing the hierarchical policy $\pi(a|s)$.
- **Dynamic Neuron Perturbation Environment:** An interface that constructs the state s_t, translates the agent's action a_t into a concrete perturbation, and computes the resulting reward R_{t+1}.
- **Adaptive Perturbation Localization Module:** The mechanism that translates the abstract action a_t into the specific numerical perturbation Δ_{dyn}, utilizing macro-function networks, a learnable dynamic mask, and neuron attribution analysis (Sundararajan, Taly, and Yan 2017; Ancona et al. 2017).
- **LLM Response Evaluation Module:** A component that assesses the quality of the corrected output y_c to generate the reward signal R_{t+1}, which in turn drives the learning of both the RL agent and the adaptive mask.

During training, this system operates in a closed loop: for each "bad case" x, the agent observes the state, selects an action to guide a perturbation, receives a reward based on the corrected output, and updates its policy. This dynamic, learning-based approach stands in contrast to static model editing methods (Yao et al. 2023), which compute a single, permanent parameter update ΔW^* based on a fixed optimization objective:

$$\Delta W^* = \arg\min_{\Delta\tilde{W}} \Big(\mathcal{L}_{edit}\big((W + \Delta\tilde{W})K_1, V_1\big) + \lambda\, \mathcal{L}_{preserve}(\Delta\tilde{W}K_0) \Big), \quad (2)$$

where K_1, V_1 represent target knowledge and K_0 represents knowledge to be preserved. The inherent static nature of these methods makes them struggle with context-dependent hallucinations, thereby highlighting the necessity of our dynamic approach.

Dynamic Neuron Perturbation Environment

The agent interacts with the LLM through a purpose-built environment, which provides a state representation $s_t \in S$ at each timestep t. The state vector s_t is a concatenation of four components: **Input Context Embedding** ($Emb(x)$), which encodes the semantic features of the input; **Baseline Model Performance** ($Scores_{baseline}$), a set of metrics (e.g., hallucination, relevance, fluency) for the LLM's unperturbed output y_h; **Current Best Performance** ($Scores_{best}$), which tracks the highest-quality scores achieved so far within the episode; and a **Normalized Step Count** ($Steps_{norm}$), which indicates the progress of the interaction. Thus, $s_t = \text{concat}(Emb(x), Scores_{baseline}, Scores_{best}, Steps_{norm})$.

From state s_t, the agent selects an action $a_t \in A$ from a hierarchical discrete action space. This decouples the decision into two levels:

- **High-Level Action ($a_H \in A_H$):** Selects a macro-level target, specifically a network category C_k from a predefined set $A_H = \{C_1, C_2, \ldots, C_{N_H}\}$ that corresponds to functional clusters of neurons.
- **Low-Level Action ($a_L = (a_L^{type}, a_L^{mag})$):** Given the high-level choice a_H, this action specifies the fine-grained intervention details: the **Perturbation Type** ($a_L^{type} \in \{\text{noise, zero, scale, } \ldots\}$) and the **Perturbation Magnitude** ($a_L^{mag} \in \{m_1, m_2, \ldots, m_{N_M}\}$).

The complete action is the tuple $a_t = (a_H, a_L^{type}, a_L^{mag})$. Upon execution of a_t, the environment facilitates the state transition $P(s_{t+1}|s_t, a_t)$. The perturbation Δ_{dyn} defined by a_t is applied to the LLM's activations, leading to a new output y_c. This output is evaluated to yield current scores, $Scores_{current}$. The environment then updates $Scores_{best}$ and increments $Steps_{norm}$ to form the next state s_{t+1}. This transition is near-deterministic, with stochasticity arising primarily from the LLM's decoding process.

The environment provides a scalar reward $R_t = R(s_t, a_t, s_{t+1})$ calculated as:

$$R_t = w_h \cdot \Delta\text{Score}_{h,t} + w_r \cdot \Delta\text{Score}_{r,t} + w_f \cdot \Delta\text{Score}_{f,t} + R_{\text{exp},t} \tag{3}$$

Here, $\Delta Score_{h,t} = Score_{h,baseline} - Score_{h,current,t}$ quantifies the reduction in hallucination. Concurrently, $\Delta Score_{r,t}$ and $\Delta Score_{f,t}$ measure the change in relevance and fluency, respectively. The weights w_h, w_r, w_f balance these competing objectives. An exploration bonus, $R_{exp,t} > 0$, is added to incentivize discovering new strategies, particularly when the current action fails to improve upon the best-known hallucination score.

Hierarchical Reinforcement Learning Agent

We employ a hierarchical reinforcement learning (HRL) framework powered by Proximal Policy Optimization (PPO), an algorithm selected for its sample efficiency and robust training stability in complex decision-making domains. The agent's architecture is bifurcated into two tiers, i.e., high-level and low-level, each implemented with its own actor (policy) and critic (value) networks to effectively manage macro and micro decisions.

The **high-level component** governs strategic, macro-level choices. Its policy network, $\pi_H(a_H|s;\theta_{\pi_H})$, is an MLP that maps the state s to a probability distribution over the macro target categories $a_H \in A_H$. The corresponding value network, $V_H(s;\theta_{V_H})$, also an MLP, estimates the expected cumulative return from state s.

$$\begin{aligned} \pi_H(A_H \mid s;\theta_{\pi_H}) &= \underbrace{\text{Softmax}\big(\text{MLP}_{\pi_H}(s;\,\theta_{\pi_H})\big)}_{\text{High-Level Policy Network}} \\ V_H(s;\theta_{V_H}) &= \underbrace{\text{MLP}_{V_H}(s;\,\theta_{V_H})}_{\text{High-Level Value Network}} \end{aligned} \tag{4}$$

Theoretical details of our hierarchical PPO are in Appendix A. and hyperparameter settings are in Appendix H. The **low-level component** makes tactical, micro-level decisions under the guidance of the high-level action. Its policy network, $\pi_L(a_L|s, a_H;\theta_{\pi_L})$, takes both the state s and the chosen high-level action a_H (via its embedding) as input. It then outputs a probability distribution over the specific perturbation types and magnitudes $a_L \in A_L$. The low-level value network, $V_L(s, a_H;\theta_{V_L})$, estimates the expected return for being in state s having committed to macro action a_H.

$$\begin{aligned} \pi_L(A_L \mid s, a_H;\theta_{\pi_L}) &= \underbrace{\text{Softmax}\Big(\text{MLP}_{\pi_L}\big(\text{concat}(s,\ \text{embed}(a_H));\theta_{\pi_L}\big)\Big)}_{\text{Low-Level Policy Network}} \\ V_L(s, a_H;\theta_{V_L}) &= \underbrace{\text{MLP}_{V_L}\Big(\text{concat}(s,\ \text{embed}(a_H));\theta_{V_L}\Big)}_{\text{Low-Level Value Network}} \end{aligned} \tag{5}$$

The **learning process** consists of two phases: experience collection and network updates. In each episode, the agent executes the combined action $a_t = (a_{H,t}, a_{L,t})$ determined by its policies, receives a reward R_t, and transitions to the next state s_{t+1}. The resulting experience tuple, $(s_t, a_{H,t}, a_{L,t}, R_t, s_{t+1}, \log \pi_H(a_{H,t} \mid s_t), \log \pi_L(a_{L,t} \mid s_t, a_{H,t}))$, is stored in separate high- and low-level replay buffers. During the update phase, both policy layers are optimized using PPO's clipped surrogate objective:

$$L^{\text{CLIP}}(\theta_\pi) = -\mathbb{E}_t\left[\min\left(r_t(\theta_\pi)\hat{A}_t,\ \text{clip}(r_t(\theta_\pi), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \tag{6}$$

where $r_t(\theta_\pi) = \frac{\pi_{\theta_\pi}(a_t|s_t)}{\pi_{\theta_{\pi,\text{old}}}(a_t|s_t)}$ is the importance sampling ratio. The advantage $\hat{A}_t$ is estimated using Generalized Advantage Estimation (GAE) (Schulman et al. 2018). The total loss function also includes a squared-error value loss $L^{\text{VF}}(\theta_V)$ and a policy entropy term $S[\pi_{\theta_\pi}(s_t)]$ to encourage exploration and prevent premature policy convergence.

Adaptive Masking and Neuron-level Causal Trace

A core challenge in our framework is translating the agent's abstract action into a precise and minimally invasive neuron-level intervention. We address this through a two-stage adaptive masking mechanism, which integrates insights from neuron-level causal tracing (see Figure 2) and shares mask definitions with prior work like llm-localization (AlKhamissi et al. 2025).

The process begins when the high-level policy selects a macro-functional network category C_k (e.g., C_{lang} for a "Language Network"). This provides a semantically meaningful, high-level target for intervention. The core of our mechanism then unfolds in two stages:

1. Learning a General Sparse Mask. For each category C_k and layer l, we introduce a **learnable dynamic mask**, $M_{k,l}$, parameterized by $\theta_{k,l}$. This mask learns a general, input-agnostic pattern of which neurons are most often relevant for correction within that functional block. It applies a gating function to produce a selection strength $M_{k,l}(i) \in [0,1]$ for each neuron i: $M_{k,l}(i;\theta_{k,l},\tau_{gate}) = \sigma(\theta_{k,l}(i)/\tau_{gate})$ The mask parameters for the entire model, $\theta_{mask} = \{\theta_{k,l}\}$, are trained via an independent optimizer to

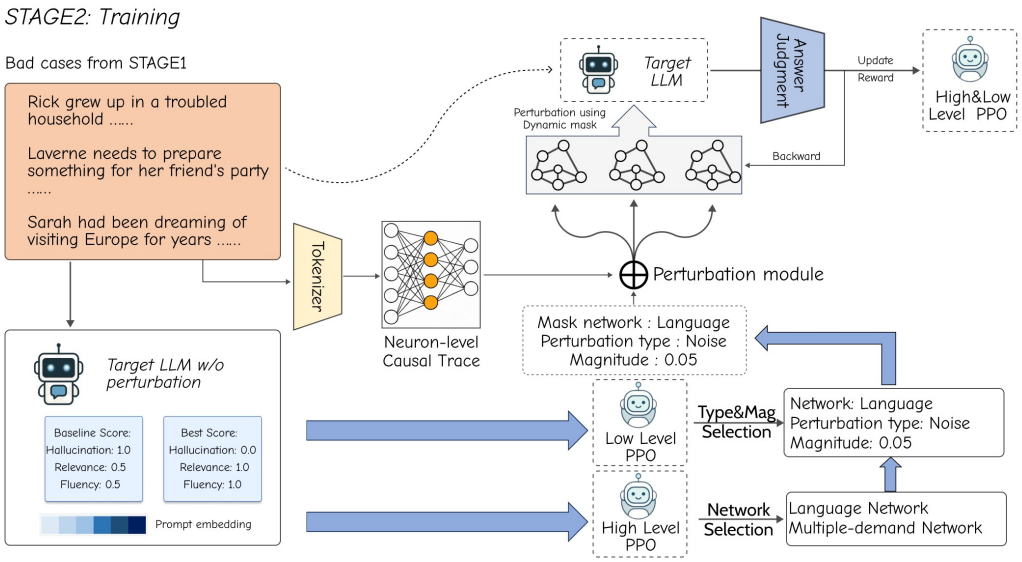

Figure 2: Stage 2: Training "bad cases from Stage 1 undergo neuron-level causal tracing to generate perturbation masks on the LLM's representations; the perturbed inputs are re-evaluated to produce a reward for optimizing two PPO agents.

minimize the loss function $\mathcal{L}_{mask}$:

$$\mathcal{L}_{mask}(\theta_{mask}) = \underbrace{-\mathbb{E}_{\text{episode}}\left[R_{total\_ep}\right]}_{\text{Negative Expected Reward}} + \underbrace{\lambda_{sparse}\sum_{k,l}\|M_{k,l}\|_1}_{\text{L1 Sparsity Penalty}} + \underbrace{\lambda_{L0}\sum_{k,l}\left\|\mathbb{I}(M_{k,l} > \epsilon_{th})\right\|_0}_{\text{L0 Sparsity Penalty}} \quad (7)$$

This objective forces the mask to be both effective (by maximizing the total episode reward R_{total_ep}) and sparse. Sparsity is enforced by the L1 and approximate L0 regularization terms, which penalize the magnitude and number of active neurons, respectively, thereby minimizing potential interference.

2. Input-Specific Adaptation. To tailor the intervention to the current input, a **Neuron-level Causal Trace module** computes neuron attribution scores, $Attr_l(x) \in \mathbb{R}^{d_l}$, for each layer l using methods like Integrated Gradients. These scores represent the "critical activation patterns" specific to input x. The final **operational mask**, $M_{op,k,l}$, is produced by dynamically modulating the general mask with these real-time attribution scores:

$$M_{op,k,l}(i) = M_{k,l}(i;\theta_{k,l}) \odot \text{normalize}(|\text{Attr}_l(x,i)|) \quad (8)$$

where normalize$(\cdot)$ scales the absolute attribution values to a $[0,1]$ range and $\odot$ denotes element-wise multiplication. This two-stage approach, learning a general sparse template and then adapting it with input-specific causal information, enables targeted, real-time perturbations without the need to retrain θ_{mask} for every new input.

Model Output Evaluation and Feedback Mechanism

The efficacy of the RL agent is critically dependent on a high-quality feedback signal. This is provided by the model output evaluation module (referenced in Figure 1 as "Answer Judgment"), which assesses the target LLM's corrected output y_c along three dimensions: hallucination (H), relevance (R), and fluency (F). The resulting numerical scores, $\{Score_H, Score_R, Score_F\}$, are essential inputs for calculating the agent's reward R_t and for optimizing the dynamic mask parameters θ_{mask}.

For the nuanced demands of open-ended generation tasks, this evaluation primarily employs **Llama2-7B-Instruct** (Touvron and et al. 2023) as a "judging LLM." This choice aligns with the recent and growing trend of using capable LLMs as scalable evaluators (Zheng and et al. 2023). We direct the judge using a meticulously crafted prompt, $\text{prompt}_{\text{eval}}(x, y_c)$, which instructs it to score the output y_c given the original input x:

$$(\text{Score}_H, \text{Score}_R, \text{Score}_F) = \text{LLM}(\text{prompt}_{\text{eval}}(x, y_c)) \quad (9)$$

For multiple-choice tasks, the reward signal is derived more directly and objectively from task-specific metrics, such as the correctness of the selected option, as detailed in our experimental setup.

We acknowledge and proactively address the potential for inherent biases or errors in any LLM-based judge (Wang

Task	StoryCloze	SST-2	BoolQ	Winogrande	Average
Baseline	65.06	88.63	70.52	50.91	68.78
ITI	68.50	**91.38**	74.10	52.80	71.70
CAA	74.65	91.16	**74.98**	52.64	73.36
SADI	67.57	88.69	70.40	51.93	69.65
Ours	**76.04**	91.30	74.47	**52.90**	**73.68**

Table 1: Accuracy of LLM-CAS on multiple-choice tasks.

Task	TrivQA	ToxiGen	TruthfulQA (TQA)					
	EM	Tox.↓	True	Info	T×I	MC1	MC2	MC3
Baseline	41.60	49.71	66.83	**99.51**	66.50	33.41	51.07	24.76
ITI	42.80	45.27	–	–	–	34.64	51.55	25.32
CAA	43.20	49.71	71.60	83.84	60.03	34.03	**52.76**	25.62
SADI	43.50	**17.14**	74.54	93.51	69.71	34.88	52.50	25.79
Ours	**44.31**	47.63	**75.12**	94.22	**70.78**	**35.47**	51.45	**26.43**

Table 2: Performance of LLM-CAS on open-ended tasks. TQA: TruthfulQA; Tox.: Toxicity score; T×I: True×Info.

Task / Metric	**Baseline**	**LLM-CAS (Ours)**	**Imp.**
StoryCloze (Accuracy)	65.06%	**76.04%**	+10.98%
Winogrande (Accuracy)	50.91%	**52.90%**	+1.99%
TruthfulQA (True Score)	66.83%	**75.12%**	+8.29%
TruthfulQA (Info Score)	99.51%	94.22%	-5.29%
TruthfulQA (True + Info)	66.50%	**70.78%**	+4.28%

Table 3: Impact of perturbing different numbers of selected neurons on the outcomes. The Adaptive Mask dynamically adjusts both the number and positions of neurons to identify the optimal perturbation strategy.

and et al. 2023). To mitigate this risk, the evaluation prompt, $\text{prompt}_{\text{eval}}$, is iteratively refined to improve its objectivity. Furthermore, by designing the reward function to depend on score *changes* (e.g., $\Delta Score_{H,t}$) rather than absolute values, we reduce the impact of any systemic scoring bias from the judge. While a full-scale human-alignment study is beyond the scope of this work, our initial qualitative checks revealed a reasonable correlation between the judge's scores and human assessments for the error types we target. This feedback mechanism is thus structured to guide the coordinated optimization of the entire framework towards an effective and robust hallucination mitigation strategy.

Experiments

Experimental Setup: We evaluate our LLM-CAS on both Multiple-choice and Open-ended generation tasks, using a comprehensive set of datasets in each case to ensure the generalizability of LLM-CAS. To better compare our LLM-CAS with SADI (Wang, Yang, and Peng 2025), we use the same evaluation method as SADI. All experiments are conducted on eight NVIDIA A100 GPUs.

Multiple-choice Tasks For the Multiple-choice tasks, we use the Story Cloze (Mostafazadeh and et al. 2016), SST-2 (Socher et al. 2013), BoolQ, and Winogrande (Sakaguchi et al. 2019) datasets. These datasets feature between 2 and 5 answer choices, primarily focused on distinguishing "correct" from "incorrect" options. We format each question with the correct answer as a prompt, then extract the logits from the LLM's response to determine its predicted choice, which is used for scoring.

Open-Ended Generation Tasks. For open-ended generation tasks, we employ the TriviaQA (Joshi et al. 2017), ToxiGen (Hartvigsen et al. 2022), and TruthfulQA (Lin, Hilton, and Evans 2021) datasets. Detailed descriptions and data splits for each dataset are available in Appendix C. Additionally, we include the multiple-choice variant of TruthfulQA to evaluate the MC-Score. For TriviaQA, we use Exact Match as the evaluation metric to assess the capabilities of the LLM-CAS framework. For ToxiGen and TruthfulQA, we use fine-tuned LLMs to evaluate the factual correctness of generated outputs. Specifically, we use `toxigen_hatebert` (based on HateBERT and ToxiGen data) for ToxiGen; for TruthfulQA, we use `truthfulqa-truth-judge-llama2-7B` to evaluate factual correctness, and `truthfulqa-info-judge-llama2-7B` to evaluate informativeness (these judges are based on LLaMA2 (Touvron and et al. 2023) and the TruthfulQA dataset). All these models can be found on Hugging Face. They have been deployed since they are fine-tuned for judging, alleviating their hallucinations.

Target LLMs Our primary baseline model is LLaMA2-7B-CHAT (Touvron and et al. 2023). To assess the generalizability of the LLM-CAS framework, we conduct experiments across LLMs with different architectures and parameter scales. For architectural diversity, we test MISTRAL-7B (Jiang and et al. 2023) and Gemma-1.1-7b-it (Team et al. 2024).

Baseline Comparisons To better illustrate the effectiveness of our LLM-CAS, we compare it against Inference-Time Intervention (ITI) (Li et al. 2024), Contrastive Activation Addition (CAA) (Panickssery et al. 2024), and SADI (Wang, Yang, and Peng 2025).

Main results

Multiple-choice Questions Effectiveness on Multiple-Choice Tasks As shown in Table 1, LLM-CAS consistently outperforms the baseline model and all other competing methods across a variety of multiple-choice question datasets, demonstrating its effectiveness in improving the accuracy of discrete choice tasks. Unlike ITI and CAA, which apply static, vector-based perturbations to neurons, both SADI and LLM-CAS employ dynamic perturbation strategies. Moreover, the superior accuracy of LLM-CAS validates the correctness of its dynamic masking combined with a PPO-based optimization, which is a clear advantage over SADI's dynamic interventions. Notably, LLM-CAS achieves a 10.98% absolute improvement over the baseline on the Story Cloze dataset, underscoring its strong potential.

Open-ended Generation Questions LLM-CAS improves the performance on open-ended generation tasks. LLM-CAS remarkably improves performance on open-ended generation tasks. To further assess its effectiveness, we evaluate

Category	True	Info	True×Info
Factual	70.5	89.8	63.3
Referential	76.7	95.1	73.0
Cultural	79.6	98.1	78.1
Subjective	73.9	91.3	67.5
Academic	75.2	94.2	70.8
Health	74.6	97.2	72.5
Avg	**75.1**	**94.2**	**70.8**

Table 4: Accuracy (%) across categories. LLM-CAS shows significant gains.

LLM-CAS on several benchmark datasets. As illustrated in Table 2, compared to the Baseline models, the LLM-CAS framework enhances the performance of LLMs across multiple dimensions, including knowledge-intensive QA (TriviaQA), safety/toxicity control (ToxiGen), and truthfulness (TruthfulQA). Notably, LLM-CAS improves the accuracy on TriviaQA by 2.71 points (from 41.60 to 44.31), and reduces toxicity in ToxiGen by 2.08 points (toxicity score from 49.71 to 47.63). Furthermore, it significantly boosts performance on TruthfulQA, increasing the MC1 score from 33.41 to 35.47 (+2.06) and the MC2 score from 51.07 to 51.45 (+0.38). These improvements suggest that LLM-CAS not only enhances factual correctness and safety but also outperforms existing steering-based methods on certain datasets.

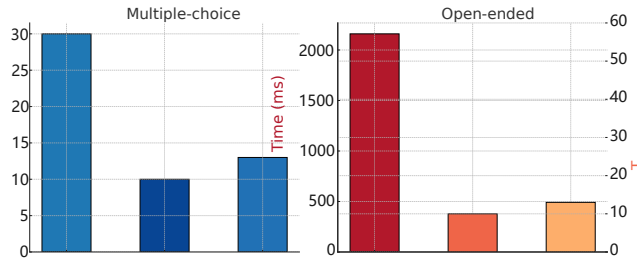

Figure 3: Comparison of PPO decision time, dynamic mask inference time, and overall model execution time. For multiple-choice tasks, "forward pass time" denotes the duration of the model's forward propagation to obtain logits;

Evaluation of LLM-CAS on Various Language Models

Although LLM-CAS achieves strong results on the widely used Llama2-7B-Chat model, its effectiveness on other LLM architectures remains unclear. To evaluate the generalizability of the LLM-CAS framework across different model families, we conduct experiments on two open-source, Transformer-based LLMs, i.e., Mistral-7B-Instruct-v0.3 and Gemma-1.1-7b-it. The results, shown in Table 5, indicate that on both Mistral and Gemma, LLM-CAS consistently improves accuracy over the baseline on the StoryCloze, SST-2, and Winogrande datasets, demonstrating its robustness to architectural variation. Notably, the largest gains are observed on StoryCloze for both models, suggesting that the neuron-level interventions provided by LLM-CAS are particularly effective for tasks closely tied to narrative coherence.

Model	StoryCloze	SST-2	Winogrande
Mistral-7B-Instruct-v0.3	21.51	90.33	58.96
Gemma-1.1-7b-it	60.95	74.08	48.46
Mistral-7B w/ LLM-CAS	34.41	90.45	59.80
Gemma-1.1 w/ LLM-CAS	69.76	79.19	49.71

Table 5: Performance of different models on various tasks with and without LLM-CAS

Ablation Studies

Variant	SST-2	BoolQ	Wino.	Story.	Avg.
LLM-CAS (Full)	**91.30**	**74.47**	**52.90**	**76.04**	**73.68**
Random mask	86.73	67.10	51.32	70.20	68.84
Random action	82.45	64.32	49.15	66.87	65.70
Rand. mask & action	80.18	62.05	47.98	63.41	63.41

Table 6: Ablation results of LLM-CAS on multiple-choice tasks. Each row removes a key component.

Dynamic Masking is Critical. Removing the dynamic masking mechanism and replacing it with random neuron selection leads to a noticeable drop in performance across multiple tasks, e.g., a decrease of 5.84 points on StoryCloze and 7.37 points on BoolQ. This demonstrates that adaptively identifying task-relevant neurons during inference is crucial for accurate decision-making. **PPO Optimization Enhances Adaptability.** Replacing PPO with non-adaptive optimization (i.e., random action selection) results in a substantial performance decline, e.g., –9.17 points on StoryCloze and –10.15 points on BoolQ. This confirms that PPO's policy, gradient approach better navigates the complex activation space of transformer-based models, enabling more effective control. **Combined Effect Exceeds the Sum.** Removing both dynamic masking and PPO causes the largest drop, reducing average accuracy to 63.41 (-10.27 vs. full). This suggests strong synergy: masking enables targeted neural modulation, while PPO learns robust control policies for adaptation.

Conclusion

This paper presents LLM-CAS, a dynamic neuron-perturbation framework that uses hierarchical reinforcement learning to apply temporary, context-aware tweaks during inference, i.e., correcting hallucinations in real-time without harming general model behavior. By combining adaptive masking with neuron-level causal tracing, it precisely targets only the activations that cause errors. Framed as an RL problem balancing factuality, relevance, and fluency, our LLM-CAS outperforms both static edits and other dynamic schemes across multiple classification and generation benchmarks. Ablation studies confirm that its dynamic masking and PPO-based policy are both essential for robust correction.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62276283, in part by the China Meteorological Administration's Science and Technology Project under Grant CMAJBGS202517, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2023A1515012985, in part by Guangdong-Hong Kong-Macao Greater Bay Area Meteorological Technology Collaborative Research Project under Grant GHMA2024Z04, in part by Fundamental Research Funds for the Central Universities, Sun Yat-sen University under Grant 23hytd006, and in part by Guangdong Provincial High-Level Young Talent Program under Grant RL2024-151-2-11.

References

AlKhamissi, B.; Tuckute, G.; Bosselut, A.; and Schrimpf, M. 2025. The LLM Language Network: A Neuroscientific Approach for Identifying Causally Task-Relevant Units. arXiv:2411.02280.

Ancona, M.; Ceolini, E.; Öztireli, A. C.; and Gross, M. H. 2017. A unified view of gradient-based attribution methods for Deep Neural Networks. *CoRR*, abs/1711.06104.

Askell, A.; Bai, Y.; Chen, A.; and et al. 2021. A General Language Assistant as a Laboratory for Alignment. *CoRR*, abs/2112.00861.

Barto, A. G.; and Mahadevan, S. 2003. Recent Advances in Hierarchical Reinforcement Learning. *Discrete Event Dynamic Systems*, 13(4): 341–379.

Black, K.; Janner, M.; Du, Y.; Kostrikov, I.; and Levine, S. 2024. Training Diffusion Models with Reinforcement Learning. arXiv:2305.13301.

Brown, T. B.; and et al., B. M. 2020. Language Models are Few-Shot Learners. *CoRR*, abs/2005.14165.

Casper, S.; Davies, X.; and et al. 2023. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback. arXiv:2307.15217.

Christiano, P.; Leike, J.; Brown, T. B.; Martic, M.; Legg, S.; and Amodei, D. 2023. Deep reinforcement learning from human preferences. arXiv:1706.03741.

Dai, D.; Dong, L.; Hao, Y.; Sui, Z.; and Wei, F. 2021. Knowledge Neurons in Pretrained Transformers. *CoRR*, abs/2104.08696.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. arXiv:2305.14314.

Fan, Y.; Hong, Y.; Wang, Q.; Bao, J.; Jiang, H.; and Song, Y. 2024. Preference-Oriented Supervised Fine-Tuning: Favoring Target Model Over Aligned Large Language Models. arXiv:2412.12865.

Farquhar, S.; Kossen, J.; Kuhn, L.; and Gal, Y. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630: 625–630.

Hartvigsen, T.; Gabriel, S.; Palangi, H.; Sap, M.; Ray, D.; and Kamar, E. 2022. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *ACL*, 3309–3326. Dublin, Ireland: Association for Computational Linguistics.

Hase, P.; Diab, M.; Celikyilmaz, A.; Li, X.; Kozareva, Z.; Stoyanov, V.; Bansal, M.; and Iyer, S. 2023. Methods for Measuring, Updating, and Visualizing Factual Beliefs in Language Models. In Vlachos, A.; and Augenstein, I., eds., *ACL*, 2714–2731. Dubrovnik, Croatia: Association for Computational Linguistics.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *CoRR*, abs/2106.09685.

Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; and Liu, T. 2025. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Transactions on Information Systems*, 43(2): 1–55.

Jiang, A. Q.; and et al., A. S. 2023. Mistral 7B. arXiv:2310.06825.

Joshi, M.; Choi, E.; Weld, D.; and Zettlemoyer, L. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In Barzilay, R.; and Kan, M.-Y., eds., *ACL*, 1601–1611. Vancouver, Canada: Association for Computational Linguistics.

Kang, K.; Wallace, E.; Tomlin, C.; Kumar, A.; and Levine, S. 2024. Unfamiliar Finetuning Examples Control How Language Models Hallucinate. arXiv:2403.05612.

Kulkarni, T. D.; Narasimhan, K.; Saeedi, A.; and Tenenbaum, J. B. 2016. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. *CoRR*, abs/1604.06057.

Li, K.; Patel, O.; Viégas, F.; Pfister, H.; and Wattenberg, M. 2024. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model. arXiv:2306.03341.

Li, X.; Zhang, J.; and Safara, F. 2021. Improving the Accuracy of Diabetes Diagnosis Applications through a Hybrid Feature Selection Algorithm. *Neural Process. Lett.*, 55(1): 153–169.

Lin, S.; Hilton, J.; and Evans, O. 2021. TruthfulQA: Measuring How Models Mimic Human Falsehoods. *CoRR*, abs/2109.07958.

McKenna, N.; Li, T.; Cheng, L.; Hosseini, M. J.; Johnson, M.; and Steedman, M. 2023. Sources of Hallucination by Large Language Models on Inference Tasks. arXiv:2305.14552.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2023a. Locating and Editing Factual Associations in GPT. arXiv:2202.05262.

Meng, K.; Sharma, A. S.; Andonian, A.; Belinkov, Y.; and Bau, D. 2023b. Mass-Editing Memory in a Transformer. arXiv:2210.07229.

Mitchell, E.; Lin, C.; Bosselut, A.; Finn, C.; and Manning, C. D. 2021. Fast Model Editing at Scale. *CoRR*, abs/2110.11309.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T. P.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous Methods for Deep Reinforcement Learning. *CoRR*, abs/1602.01783.

Mnih, V.; and Kavukcuoglu, K. e. a. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Mostafazadeh, N.; and Chambers, N. e. a. 2016. A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. In Knight, K.; Nenkova, A.; and Rambow, O., eds., *ACL*, 839–849. Association for Computational Linguistics.

Mostafazadeh, N.; and et al., N. C. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *CoRR*, abs/1604.01696.

Panickssery, N.; Gabrieli, N.; Schulz, J.; Tong, M.; Hubinger, E.; and Turner, A. M. 2024. Steering Llama 2 via Contrastive Activation Addition. arXiv:2312.06681.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI. OpenAI Blog.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2019. WINOGRANDE: An Adversarial Winograd Schema Challenge at Scale. *CoRR*, abs/1907.10641.

Schulman, J.; Levine, S.; Moritz, P.; Jordan, M. I.; and Abbeel, P. 2015. Trust Region Policy Optimization. *CoRR*, abs/1502.05477.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2018. High-Dimensional Continuous Control Using Generalized Advantage Estimation. arXiv:1506.02438.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Yarowsky, D.; Baldwin, T.; Korhonen, A.; Livescu, K.; and Bethard, S., eds., *EMNLP*, 1631–1642. Seattle, Washington, USA: Association for Computational Linguistics.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. *CoRR*, abs/1703.01365.

Team, G.; Mesnard, T.; Hardin, C.; and et al., R. D. 2024. Gemma: Open Models Based on Gemini Research and Technology. arXiv:2403.08295.

Touvron, H.; and et al., L. M. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Wang, P.; and et al., L. L. 2023. Large Language Models are not Fair Evaluators. arXiv:2305.17926.

Wang, W.; Yang, J.; and Peng, W. 2025. Semantics-Adaptive Activation Intervention for LLMs via Dynamic Steering Vectors. In *ICLR*.

Yao, J.; Zhang, J.; Pan, X.; Wu, T.; and Xiao, C. 2024. DepthSSC: Monocular 3D Semantic Scene Completion via Depth-Spatial Alignment and Voxel Adaptation. arXiv:2311.17084.

Yao, Y.; Wang, P.; Tian, B.; Cheng, S.; Li, Z.; Deng, S.; Chen, H.; and Zhang, N. 2023. Editing Large Language Models: Problems, Methods, and Opportunities. arXiv:2305.13172.

Zhang, J.; Cai, K.; Fan, Y.; Liu, N.; and Wang, K. 2025a. MAT-Agent: Adaptive Multi-Agent Training Optimization. In *NeurIPS*.

Zhang, J.; Cai, K.; Fan, Y.; Wang, J.; and Wang, K. 2025b. CF-VLM:CounterFactual Vision-Language Fine-tuning. arXiv:2506.17267.

Zhang, J.; Cai, K.; Guo, X.; Liu, S.; Lv, Q.; Chen, R.; Yang, J.; Fan, Y.; Sun, X.; Wang, J.; Chen, Z.; Lin, L.; and Wang, K. 2025c. MM-CoT:A Benchmark for Probing Visual Chain-of-Thought Reasoning in Multimodal Models. arXiv:2512.08228.

Zhang, J.; Cai, K.; Yang, J.; Wang, J.; Tang, C.; and Wang, K. 2025d. Top-Down Semantic Refinement for Image Captioning. arXiv:2510.22391.

Zhang, J.; Cai, K.; Yang, J.; and Wang, K. 2025e. Learning Dynamics of VLM Finetuning. arXiv:2510.11978.

Zhang, J.; Cai, K.; Zeng, Q.; Liu, N.; Fan, S.; Chen, Z.; and Wang, K. 2025f. Failure-Driven Workflow Refinement. arXiv:2510.10035.

Zhang, J.; Fan, Y.; Cai, K.; Huang, Z.; Sun, X.; Wang, J.; Tang, C.; and Wang, K. 2025g. DrDiff: Dynamic Routing Diffusion with Hierarchical Attention for Breaking the Efficiency-Quality Trade-off. arXiv:2509.02785.

Zhang, J.; Fan, Y.; Cai, K.; Sun, X.; and Wang, K. 2025h. OSC: Cognitive Orchestration through Dynamic Knowledge Alignment in Multi-Agent LLM Collaboration. arXiv:2509.04876.

Zhang, J.; Fan, Y.; Cai, K.; and Wang, K. 2025i. Kolmogorov-Arnold Fourier Networks. arXiv:2502.06018.

Zhang, J.; Fan, Y.; Cai, K.; Yang, J.; Yao, J.; Wang, J.; Qu, G.; Chen, Z.; and Wang, K. 2026a. Why Keep Your Doubts to Yourself? Trading Visual Uncertainties in Multi-Agent Bandit Systems. arXiv:2601.18735.

Zhang, J.; Fan, Y.; Cai, K.; Yang, J.; Zheng, Y.; Lam, K.-Y.; Lin, L.; and Wang, K. 2026b. Spectral Gating Networks. arXiv:2602.07679.

Zhang, J.; Fan, Y.; Lin, W.; Chen, R.; Jiang, H.; Chai, W.; Wang, J.; and Wang, K. 2025j. GAM-Agent: Game-Theoretic and Uncertainty-Aware Collaboration for Complex Visual Reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhang, J.; Fan, Y.; Wen, Z.; Wang, J.; and Wang, K. 2025k. Tri-MARF: A Tri-Modal Multi-Agent Responsive Framework for Comprehensive 3D Object Annotation. In *NeurIPS*.

Zhang, J.; Guo, X.; Cai, K.; Lv, Q.; Fan, Y.; Chai, W.; Wang, J.; and Wang, K. 2025l. HybridToken-VLM: Hybrid Token Compression for Vision-Language Models. arXiv:2512.08240.

Zhang, J.; Huang, Z.; Fan, Y.; Liu, N.; Li, M.; Yang, Z.; Yao, J.; Wang, J.; and Wang, K. 2025m. KABB: Knowledge-Aware Bayesian Bandits for Dynamic Expert Coordination in Multi-Agent Systems. In *ICML*.

Zhang, J.; Liu, N.; Lyu, Q.; Yang, J.; and Wang, K. 2026c. Rational ANOVA Networks.

Zheng, L.; and et al., W.-L. C. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.05685.