Augmenting Multi-Agent Communication with State Delta Trajectory

Anonymous ACL submission

Abstract

Multi-agent techniques such as role playing or multi-turn debates have been shown to be effective in improving the performance of large language models (LLMs) in downstream tasks. Despite their differences in workflows, existing LLM-based multi-agent systems mostly use natural language for agent communication. While this is appealing for its simplicity and interpretability, it also introduces inevitable information loss as one model must down sample its continuous state vectors to concrete tokens before transferring them to the other model. Such losses are particularly significant when the information to transfer is not simple facts, but reasoning logics or abstractive thoughts. To tackle this problem, we propose a new communication protocol that transfers both natural language tokens and token-wise state transition trajectory from one agent to another. Particularly, compared to the actual state value, we find that the sequence of state changes in LLMs after generating each token can better reflect the information hidden behind the inference process, so we propose a State Delta Encoding (SDE) method to represent state transition trajectories. The experimental results show that multi-agent systems with SDE achieve SOTA performance compared to other communication protocols, particularly in tasks that involve complex reasoning. This shows the potential of communication augmentation for LLM-based multi-agent systems.[1]

1 Introduction

Multi-agent systems based on Large Language Models (LLMs) have demonstrated remarkable performance in solving complex tasks (Taicheng Guo, 2024; Dong et al., 2024; Du et al., 2024). While it is not surprising that combining outputs from different LLMs could improve the system performance (Xu et al., 2023; Chu et al., 2024; Xu et al., 2024), studies have shown that building a multi-agent system with a single base LLM can also boost the LLM's performance (Chi-Min Chan, 2024; Hong et al., 2024; Du et al., 2024). These systems construct multiple agents from the same LLM, varying their profiles or access to information, which can be seen as another form of the inference scaling law (Chen et al., 2024; Qian et al., 2025). Therefore, how to build effective multi-agent frameworks or workflows to improve LLMs in downstream tasks have been widely studied in recent literature.

Despite their differences in motivation and methodology, the majority of existing multi-agent frameworks rely on natural language tokens to build the communication protocol between agents (Wu et al., 2024; Li et al., 2023a; Qian et al., 2024; Xie et al., 2024), which may not be the optimal solution for agent communication. Natural language is appealing for its generalizability and interpretability, but it down samples the model's internal states to concrete tokens before transferring information, which could lead to information loss in many cases. For example, in inference, an LLM may consider multiple reasoning paths, in both correct and incorrect ones could appear. However, only one path is ultimately sampled and presented to other agents (Yu et al., 2024; Zhou et al., 2024), and if the sampled one is incorrect, there is no way for other agents to recover what is lost in this sampling process.

Intuitively, when agents are built from a single base LLM (i.e., a single-LLM-based multi-agent system), information loss from natural language seems unnecessary because all agents are sharing the same semantic and parametric space created by the base LLM. For example, a straightforward solution to mitigate the information loss problem above is to transfer not just the final tokens, but also the token probabilities and weighted token embeddings to the other agents (Pham et al., 2024). Yet, these

[1] We have open-sourced all the code and data in `https://anonymous.4open.science/r/StateDeltaEncoding/`.

methods produce marginal improvements over natural language methods empirically, which indicates that simply modeling output probability distributions is not enough to convey important information hidden in the inference process of an LLM-based agent. Thus, finding the best way to convey internal reasoning information from one agent to another has become a key research question for the studies of multi-agent communication protocols.

In this paper, we propose to augment single-LLM-based multi-agent communication directly with the model's internal states. Particularly, as different agents often have different initial prompts or local context in existing multi-agent frameworks, we believe that directly transferring the internal state sequence, which we refer to as the state transition trajectory, from one agent to another may not be feasible. Instead, inspired by the idea of delta encoding (Mogul et al., 1997; Burns and Long, 1997), we propose to transfer information between agents based on both natural language tokens and the sequence of changes in the agent's internal states, which we refer to as the State Delta Encoding (SDE). When one agent is generating output tokens, SDE records the differences between the hidden states of adjacent tokens. Then, when another agent is encoding these output tokens, SDE adds the trajectory of these differences (i.e., state deltas) to the corresponding tokens in order to recover the information lost in token sampling. Our experiments on information asymmetry tasks (e.g., QA with unshared resources (Dhingra et al., 2017; Geva et al., 2021; Talmor and Berant, 2018)) and information symmetry tasks (e.g., debates (Du et al., 2024) and agent workflows (Yao et al., 2023)) show that SDE can significantly improve the performance of multi-agent systems. The advantages of SDE are particularly strong on tasks that involve complicated logic reasoning rather than simple fact communication. This demonstrates the potential of multi-agent communication protocols beyond natural language and could lead to multiple research directions in future studies.

In summary, the contributions of our paper are as follows:

- We propose SDE, a novel multi-agent communication protocol that augments natural language with LLM's hidden states, bridging the gap between surface-level communication and latent reasoning.
- We introduce the concept of state delta, which captures the reasoning process hidden behind output tokens and can serve as an effective medium to transfer information among single-LLM-based agents.
- We evaluate existing communication protocols and SDE on both information asymmetry and symmetry tasks. The results show that SDE achieves state-of-the-art performance and outperforms prior methods by up to 17.3% in tasks that require complex reasoning.

2 Related Work

2.1 LLM based Multi-Agent System

Recent advances have shown that coordinating multiple LLM-based agents allows stronger performance in tasks such as software development (Qian et al., 2024), world simulations (Park et al., 2023; Li et al., 2024), and embodied problem solving (Zhang et al., 2024).

While some systems employ diverse LLMs to combine their strengths and mitigate individual biases (Chu et al., 2024; Xu et al., 2023), many works adopt a single LLM to construct all agents, varying their behavior through different profiles or access to distinct information (Qian et al., 2024; Li et al., 2023a). We refer to these as single-LLM-based multi-agent systems. Such systems have demonstrated effectiveness through structured interactions like debates (Du et al., 2024) and task-specific workflows (Wu et al., 2024; Qian et al., 2024), benefiting from the scale of the inference process (Chen et al., 2024; Qian et al., 2025). Our work focuses on optimizing this type of systems and aims to make better use of each inference step during inter-agent communication.

2.2 Multi-agent Communication

Most LLM-based agent systems use natural language for communication (Li et al., 2023a; Wu et al., 2024; Chi-Min Chan, 2024). While natural language offers flexibility, it may also introduce potential information loss.

A recent attempt to address this issue, CIPHER (Pham et al., 2024), replaces natural language tokens with probability-weighted token embeddings during agent communication, showing potential in multi-agent debate settings. However, this approach only leverages surface-level token probability distributions from the final output layer, overlooking deeper, more informative, and more valuable hidden representations.

Another approach (Ramesh and Li, 2025) attempts to directly transfer hidden states between agents, but is restricted to a unidirectional transfer, where hidden states from a text-reading agent are transferred to an output-generating agent. It does not support dynamic, interactive exchanges typical in multi-agent systems.

Building upon these insights, our method utilizes the dynamics of hidden states during inference and supports any inter-agent communication.

2.3 Latent Space Arithmetic

Recent studies have explored controlling the outputs of frozen LLMs by manipulating their hidden states during inference (Li et al., 2023b; Subramani et al., 2022). Several approaches have proposed extracting steering vectors to manipulate the quality (Li et al., 2023b; Subramani et al., 2022; Rimsky et al., 2024) or semantic direction (Turner et al., 2024) of model outputs. For example, ActAdd (Turner et al., 2024) derives steering vectors by computing hidden state differences under prompts with or without a special keyword, and adds these vectors during inference to guide generations to a desired direction.

Inspired by these works, we also manipulate intermediate representations at inference time. Rather than operating within a single model, we extract internal states from one agent and inject them into another. This cross-agent state sharing aims to enhance mutual understanding and coordination in multi-agent systems.

3 Methodology

We present a novel communication protocol for single-LLM-based multi-agent systems, which is constructed using a method we call **State Delta Encoding (SDE)**. Rather than replacing natural language, SDE augments it by transferring token-wise changes of hidden states, providing richer reasoning traces. This section introduces SDE as a state representation mechanism and describes how we use it to build a new communication protocol. The protocol with SDE is illustrated in Figure 1.

We focus on the multi-agent systems in which all agents are constructed from the same transformer-based language model. Consider two agents, Alice and Bob. Alice receives an input and generates a response output_A, which is a sequence of natural language tokens $t_1, t_2, t_3, \cdots, t_n$. In natural language communication, output_A is inserted directly into the input prompt of Bob. Formally, the prompt received by Bob, denoted as prompt_B, takes the form $\{\mathrm{X}\ \mathrm{output}_A \mathrm{Y}\}$, where X and Y are additional contexts such as task instructions, environmental information, and responses from other agents. Bob then generates conditioned on prompt_B. However, due to sampling, the token sequence output_A reflects only a single reasoning path chosen by Alice, making it difficult for Bob to understand Alice's full intentions.

The inference process in causal LLMs is repeatedly performing forward propagation based on the input prompt and previously generated tokens $t_1, t_2, \cdots, t_{i-1}$ to predict the next token t_i. When Alice generates token t_i in $\mathbf{output}_A$, let the hidden states $h^l_{A,i}$ denote the output of the l_{th} transformer layer in the language model. Each $h^l_{A,i}$ is a vector representing the contextualized embedding of t_i, conditioned on the input prompt and previously generated tokens. We define the state trajectory at layer l during Alice's generation as the ordered sequence of hidden states:

$$\mathcal{H}^l_A = \{h^l_{A,0}, h^l_{A,1}, \cdots, h^l_{A,n}\} \tag{1}$$

Here, $h^l_{A,0}$ refers to the hidden states corresponding to the last token of Alice's input prompt, serving as the initial states before generation.

As discussed in Section 1, to prevent Bob's generation from being interfered with Alice's profile or local contexts, we avoid directly transferring the original states trajectory $\mathcal{H}^l_A$. Instead, inspired by the idea of delta encoding (Mogul et al., 1997; Burns and Long, 1997), we compute the differences between successive hidden states for each generated token, and define the state delta trajectory as follows:

$$\mathcal{S}^l_A = \{s^l_1, s^l_2, \cdots, s^l_n\}, \text{ where } s^l_i = h^l_{A,i} - h^l_{A,i-1} \tag{2}$$

Each s^l_i, referred to as a state delta, represents the internal change associated with the generation of token t_i. The state delta trajectory serves as a context-agnostic trace of the reasoning dynamics within the LLM. This process is called State Delta Encoding (SDE).

During communication, the state deltas serve as auxiliary information to improve Bob's understanding of the natural language response output_A. Inspired by the use of steering vectors (Turner et al., 2024), we treat each state delta as a steering vector and add it directly to the corresponding hidden states. Formally, recall that $\mathrm{prompt}_B =$

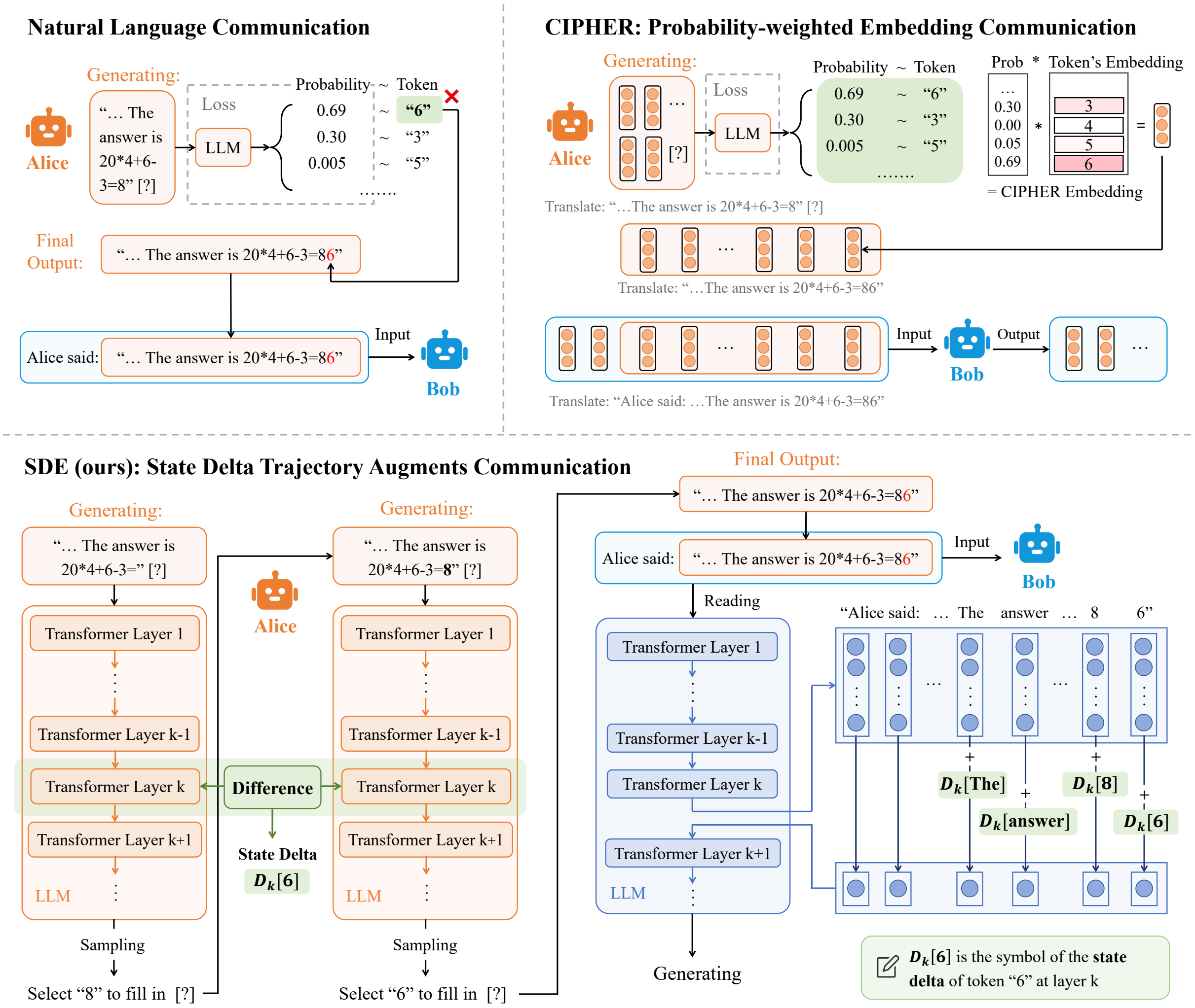

Figure 1: Comparison of different communication protocols in single-LLM-based multi-agent systems. **Top-left:** Natural language communication may introduce information loss due to sampling, thereby leading to incorrect claims being transferred. **Top-right:** CIPHER (Pham et al., 2024) improves by transferring probability-weighted embeddings instead of tokens, but still lacks deeper reasoning representations. **Bottom:** Our protocol (SDE) augments natural language tokens with the difference between hidden states of two adjacent tokens (state delta), bridging the gap between surface-level communication and latent reasoning.

$\{X \text{ output}_A \ Y\} = \{X, t_1, t_2, \cdots, t_n, Y\}$. When Bob processes output_A for generation, we inject the corresponding state deltas trajectory $\mathcal{S}_A^l$ into the hidden states at layer l before passing them to the next layer, in order to augment each natural language token. The hidden states $h_{B,j}^l$ of the token at position j in prompt_B are updated as follows:

$$h_{B,j}^l{}' = \begin{cases} h_{B,j}^l + s_i^l & \text{the position of } t_i \text{ is } j \\ h_{B,j}^l & \text{otherwise} \end{cases} \quad (3)$$

The modified hidden states $h_B^l{}'$ are passed to the layer $l+1$ for continued inference. In this way, Bob not only receives the tokens, but also accesses the latent trace of Alice's internal decision-making process. This communication protocol avoids overwriting Bob's own reasoning while guiding it to better understand Alice's generation trajectory.

Layer selection. To minimize the impact on the model's generation capabilities, we apply SDE to only a few carefully selected layers. The optimal layers for injection depend on the model's architecture and scale, but once selected, they work well across various downstream tasks, which indicates that the selection is largely task-agnostic. Layer selection is performed via a simple preliminary experiment and remains fixed for each model throughout all subsequent tasks. Details of our selection process are provided in Section 4.5.

4 Experimental Setup

We evaluate our approach in two settings: (1) the information asymmetry (IA) setting, where agents have access to different sets of knowledge and must collaborate to answer a question; and (2) the information symmetry (IS) setting, including multi-agent debates and agent workflows, where all agents share the same information. More implementation details are provided in Appendix B.

4.1 Information Asymmetry (IA) Tasks

To simulate the cooperation process of multi-agent systems with information gaps, we propose to construct a set of information asymmetry (IA) tasks where each agent possess a unique set of information (i.e., documents) and the target task can be finished better through the collaboration of all agents.

Specifically, we build such tasks on several factual QA benchmarks that require the system to retrieve multiple relevant documents to answer a question. We retrieve 6 relevant documents for each question (using BM25 as a retriever) and evenly distribute them to 2 agents as private corpora. To answer a target question, the agents must ask questions and respond to the questions asked by other agents based on their private corpus in order to gather the necessary information to generate the final answer. The agents are allowed to discuss for up to 5 rounds, and the discussion ends when either agent generates a formatted answer.

Datasets. We evaluate our approach on three benchmarks of varied difficulty. (i) **Quasar-T** (Dhingra et al., 2017) consists of simple knowledge questions collected from various sources on the Internet. (ii) **ComplexWebQuestions (CWQ)** (Talmor and Berant, 2018) involves multi-hop, web-based questions, which tests the model's reasoning ability over web content. (iii) **StrategyQA** (Geva et al., 2021) contains yes / no questions that requires multi-step strategic reasoning. We use the first 300 questions of each dataset to build tasks. Each question is scored by averaging over all formatted answers. We report the average exact match (EM) scores and F1 scores in Quasar-T and ComplexWebQuestions tasks and the average accuracy in StrategyQA tasks.

4.2 Information Symmetry (IS) Tasks

To evaluate how effectively agents can communicate and refine their reasoning with full information sharing, we design a set of tasks in the information symmetry (IS) setting. We construct two types of IS tasks: multi-agent debate and agent workflows. In both types, all agents have access to the same information and are required to interact by passing and refining intermediate thoughts through different structured communication frameworks.

4.2.1 Multi-agent Debates

Inspired by Du et al., we build multi-agent debate tasks on several mathematical or logical reasoning datasets. At the beginning of a debate, each agent independently generates an initial answer to the same question. Then, in subsequent rounds, they repeatedly revise their response after observing the previous round responses of their peers. We simulate a 3-round debate involving 2 agents.

Datasets. We evaluate our approach on four datasets. (i) **GSM8K** (Cobbe et al., 2021) contains high quality grade school math problems. (ii) MMLU (Hendrycks et al., 2021) is a multiple choice benchmark covering a wide range of subjects. we use three datasets of different categories in this benchmark: mathematical datasets **Abstract Algebra**, **College Mathematics** and logical reasoning dataset **Formal Logic**. We use the first 300 questions from GSM8K and the full sets of the three subsets of MMLU to build tasks. The reported score for each question is the average accuracy of all agents' responses in the last round.

4.2.2 Agent Workflows

We adapt the ReAct (Yao et al., 2023) framework to construct multi-agent workflow tasks, where agents collaborate sequentially to solve a problem by passing along thoughts and actions. At each step, an agent produces a thought and an action based on all previous generations, and the environment returns an observation based on the action, which becomes a part of the input for the next agent. Each question is solved by up to 7 agents taking turns in sequence.

Datasets. We evaluate our approach on factual QA benchmarks and a fact verification benchmark. For question answering, we use two multi-hop question datasets: **HotpotQA** (Yang et al., 2018), **StrategyQA** (Geva et al., 2021). For fact verification, we use the **FEVER** (Thorne et al., 2018) dataset. We build tasks using the first 300 questions from each dataset. For evaluation, we report accuracy for the StrategyQA and FEVER tasks, and both EM and F1 scores for the HotpotQA task.

4.3 Baselines

We compare our proposed approach with the following three baselines:

- **Single**. The responses are generated by a single agent and are in natural language.

- **Natural Language (NL)**. For communication from Alice to Bob, the natural language tokens generated by Alice are inserted into Bob's input prompt.
- **CIPHER** (Pham et al., 2024). CIPHER extracts the probability distribution of each token of the corresponding forward pass, and uses this distribution to weight all tokens' embeddings, resulting in a CIPHER embedding. For communication from Alice to Bob, the CIPHER embedding sequences generated by Alice are inserted into Bob's input prompt in embedding form.

NL and CIPHER use the same implementation across all tasks, while Single is implemented differently in each setting to accommodate tasks. We provide scenario-specific details in Appendix B.

4.4 LLM Selection and Generation Settings

We conducted experiments on several open-source instruction-tuned LLMs. To validate the broad effectiveness of SDE, we conducted experiments with LLMs of different series on various scales, including Qwen2.5-7B-Instruct (Yang et al., 2024), Llama3.1-8B-Instruct (Meta, 2024), and Qwen2.5-14B-Instruct (Yang et al., 2024).

To ensure the reproducibility of our results in IA tasks and agent workflow tasks, both NL and SDE generate responses using greedy decoding. Since CIPHER does not involve sampling but is affected by temperature, we set the temperature to 0 for consistency. In multi-agent debate tasks, to promote diversity in the initial responses of different agents, we use the model's default sampling and temperature settings for generation, and all reported results are averaged over three independent runs. More detailed settings and prompts can be found in Appendix B and Appendix D.

4.5 Layer Selection

We identify suitable injection layers through a simple preliminary experiment. Specifically, we construct an IA task using the 2WikiMultihopQA (Ho et al., 2020) dataset, following the settings described in Section 4.1. For each model, we evaluate SDE's performance when modifying each layer on the first 300 questions. Considering model scales, we select 1, 2, or 3 layers for 7B, 8B, and 14B models, respectively. These selected layers are then used consistently across all experiments. Notably, 2WikiMultihopQA is used only for this selection procedure and not in any main evaluation. Our main results show that optimal layers depend primarily on the model itself, rather than the downstream task. Further analysis on the impact of different layer selections and layer counts is provided in Section 5.2.

Detailed results and specific layer selections are reported in Appendix A.

5 Results

5.1 Main Experiments

In this section, we present the main experimental results and an analysis of our proposed SDE with other baselines in the above three tasks. In the following, we provide a detailed analysis of our results.

Overall analysis. Multi-agent systems perform better than single agents directly answering in most cases. In particular, SDE outperforms existing communication protocols (NL and CIPHER) almost all tasks. These improvements suggest that enriching communication with hidden states can indeed enhance the final collaboration performance of multi-agent systems.

Specifically, Table 1 shows the results of IA tasks. SDE achieves a performance improvement of 0.3% to 8.9% compared to the best-performing baseline in most tasks, with particular notable improvements on the Llama-8B-Instruct model. The improvements are generally more significant on multi-hop datasets CWQ and StrategyQA compared to the simple question dataset Quasar-T, indicating that SDE is more effective in tasks requiring complex, multi-step reasoning.

For the IS setting, Table 2 shows the results of multi-agent debate tasks, where SDE enhances performance ranging from 0.3% to 13.67% compared to the best-performing baseline. In particular, there are significant improvements in complex mathematical and logical reasoning datasets of MMLU, where SDE consistently shows a great improvement across all evaluated models. Furthermore, our experiments with Qwen2.5-7B-Instruct in the agent workflow tasks (Table 3) reveal that SDE can also enhance multi-agent workflow architectures, with improvements up to 17.3%.

Analysis among different tasks. Results on the IA tasks demonstrate that SDE meets the fundamental requirements of communication — accurately and effectively delivering information. Although SDE and NL performed similarly, the superior performance of SDE compared to CIPHER also in-

Table 1: The experimental results in the information asymmetry tasks of SDE and other baselines on three benchmarks. The best results are in bold.

Model	Method	Quasar-T		CWQ		StrategyQA
		EM	F1	EM	F1	Accuracy
Qwen2.5-7B-Instruct	**Single**	0.2367	0.2791	0.2967	0.3631	0.1700
	NL	0.3050	0.3748	0.3117	0.4304	0.4433
	CIPHER	0.2817	0.3567	0.2967	0.4040	0.3733
	SDE(ours)	**0.3150**	**0.3772**	**0.3167**	**0.4444**	**0.4550**
Llama3.1-8B-Instruct	**Single**	0.2333	0.2809	0.2467	0.3239	0.1500
	NL	0.2850	0.3496	0.3250	0.4288	0.4967
	CIPHER	0.2767	0.3488	0.3417	0.4526	0.5033
	SDE(ours)	**0.3050**	**0.3665**	**0.3517**	**0.4640**	**0.5483**
Qwen2.5-14B-Instruct	**Single**	0.3267	0.3845	0.3467	0.4258	0.4533
	NL	**0.3717**	**0.4451**	0.3750	0.4967	0.6733
	CIPHER	0.3517	0.4208	0.3500	0.4837	0.6433
	SDE(ours)	**0.3717**	0.4437	**0.3817**	**0.4980**	**0.6817**

Table 2: The experimental results in the multi-agent debate tasks of SDE and other baselines on four benchmarks. Each reported result is the average of three independent runs. The best results are in bold.

Model	Method	GSM8K	Abstract Algebra	College Math	Formal Logic
Qwen2.5-7B-Instruct	**Single**	0.8789	0.4767	0.3900	0.4497
	NL	0.9061	0.4583	0.3617	0.4762
	CIPHER	0.8933	0.4850	0.3700	0.4881
	SDE(ours)	**0.9178**	**0.5167**	**0.4433**	**0.5198**
Llama3.1-8B-Instruct	**Single**	0.7867	0.2267	0.2167	0.3571
	NL	0.8328	0.2833	0.2267	0.3889
	CIPHER	0.8167	0.2150	0.1950	0.3532
	SDE(ours)	**0.8450**	**0.3017**	**0.2417**	**0.4220**
Qwen2.5-14B-Instruct	**Single**	0.9111	0.5667	0.5067	0.5661
	NL	0.9311	0.7100	0.6350	0.6085
	CIPHER	0.9300	0.6500	0.6350	0.5675
	SDE(ours)	**0.9339**	**0.7533**	**0.6950**	**0.6574**

Table 3: The experimental results in the agent workflow tasks of SDE and other baselines using Qwen2.5-7B-Instruct. The best results are in bold.

Method	FEVER	HotpotQA		StrategyQA
	Accuracy	EM	F1	Accuracy
Single	0.0067	0.1567	0.2192	0.1567
NL	0.2300	0.2100	0.3153	0.3167
CIPHER	0.1800	0.2000	0.2879	0.3267
SDE(ours)	**0.2667**	**0.2267**	**0.3196**	**0.3833**

dicates that SDE is better equipped to handle scenarios demanding higher precision in information delivery.

The more significant improvements in IS tasks indicate that SDE not only supports information delivery but also enhances agents' understanding of the reasoning processes behind the generated contents. This deeper comprehension boosts the overall performance of multi-agent collaboration.

Moreover, we compare our method on StrategyQA using the same model Qwen2.5-7B-Instruct, under two different settings: information asymmetry and agent workflows. Our results show that the agent workflow tasks — which requires more complex reasoning — benefits more significantly from our approach. This also suggests that SDE is particularly effective in tasks that involve more complex reasoning processes.

5.2 Different Layer Selections

In this section, we investigate the impact of different layer selection strategies. Following the layer selection procedure proposed in Section 4.5, we compare three strategies: a combination of top-k layers, all layers, and only the top-ranking layer.

As the experiments using Qwen2.5-14B-Instruct shown in Figure 2, modifying the combined top-k layers (where k $\leq$ 4) results in little performance

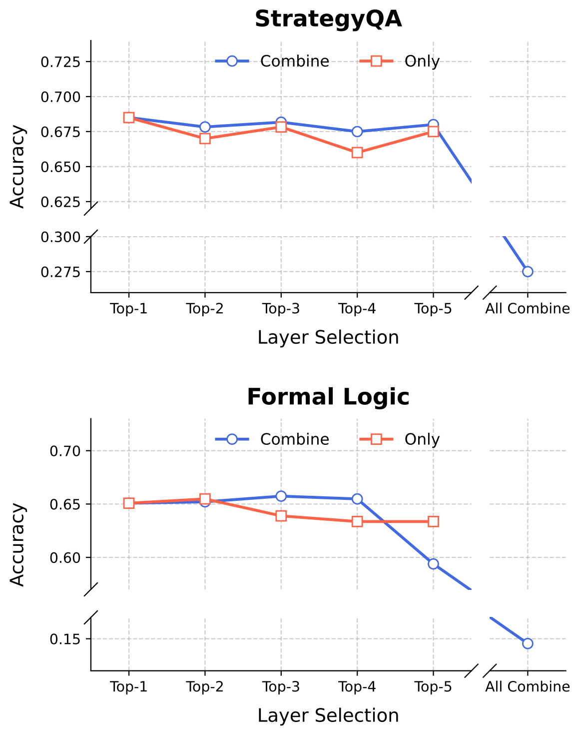

Figure 2: Ablation results for different layer selection strategies on StrategyQA (information asymmetry) and Formal Logic (multi-agent debate) tasks using Qwen2.5-14B-Instruct. We compare modifying the combined top-k layers, all layers, and only the top-k layer.

differences compared to modifying only the top-1 layer. At the same time, it offers greater stability than modifying a single layer. However, modifying all layers leads to a significant performance drop, likely due to the major interference with the model's generation capabilities. Therefore, to preserve the model's generation ability and ensure stable performance of SDE, we recommend applying the proposed layer selection procedure to the target model and modifying only a small number of top-ranking layers (e.g., 1-3). Additional experiments on other models are provided in Appendix C.

5.3 Ablation Study on State Delta

To evaluate the effectiveness of the proposed state delta, we conducted an ablation study comparing the performance of the full SDE with a variant that uses the original hidden states of each token instead of their differences.

As shown in Table 4, removing state deltas consistently leads to performance drops in all settings. Moreover, in some cases, the performance of the variant even falls below that of using natural language alone. This indicates that directly augmenting with unprocessed hidden states may introduce noise, thereby impairing the agent's reasoning.

Table 4: Ablation results on the impact of state deltas in information asymmetry tasks (Quasar-T and CWQ datasets, EM scores) and multi-agent debate tasks (College Mathematics and Formal Logic datasets). "w/o delta" denotes the variant using original hidden states. The method with better performance is bold.

		Quasar-T	CWQ	CM	FL
Q-7B	NL	0.3050	0.3117	0.3617	0.4762
	w/o delta	0.2950	0.3133	0.4033	0.4616
	SDE	**0.3150**	**0.3167**	**0.4433**	**0.5198**
L-8B	NL	0.2850	0.3250	0.2450	0.3889
	w/o delta	0.2750	0.2967	0.2467	0.3942
	SDE	**0.3050**	**0.3517**	**0.2967**	**0.4220**

Table 5: Ablation study on the Formal Logic dataset using Qwen2.5-7B-Instruct, comparing different numbers of agents (top) and different numbers of rounds (bottom) in multi-agent debate tasks.

Rounds	Agents	NL	CIPHER	SDE(ours)
3	**2**	0.4762	0.4881	**0.5198**
3	**3**	0.4489	0.4312	**0.5150**
3	**4**	0.4530	0.4365	**0.5179**
3	**5**	0.4947	0.4317	**0.5138**
2	2	0.4524	0.4881	**0.5132**
3	2	0.4762	0.4881	**0.5198**
4	2	0.4537	0.4881	**0.5225**
5	2	0.4603	0.4881	**0.5212**

5.4 Multi-agent Debate in Different Settings

To investigate how the number of agents and rounds affects the performance in the multi-agent debate tasks, we conduct an ablation study. As shown in Table 5, SDE consistently outperforms NL and CIPHER across different numbers of agents and rounds, suggesting that SDE is robust to variations in these configurations.

6 Conclusions

In this work, we propose State Delta Encoding (SDE) and use it to build a new single-LLM-based multi-agent communication protocol. By encoding token-wise hidden state changes, SDE captures the dynamic reasoning process during generation and reduces interference from irrelevant agent context. The protocol with SDE augments natural language messages with token-wise state delta trajectory, enabling richer agent communication. Experiments in both information asymmetry and symmetry tasks show that SDE outperforms existing communication protocols, especially in complex reasoning tasks. Our findings highlight the potential to improve communication beyond natural language and open new directions.

7 Limitations

While SDE shows promising improvements in multi-agent performance, it also has several limitations. First, SDE assumes that the hidden states of the generating agent can be easily extracted and injected into the receiving agent. However, this requirement may not be feasible for agents based on black-box models without internal access. Second, incorporating hidden states increases the communication bandwidth between agents, particularly for long context communication or large models. Although SDE modifies only a small number of layers, this overhead may still require compression or optimization. Future work can explore selective transmission of important states or apply compression to reduce the cost of state deltas.

References

Randal C. Burns and Darrell D. E. Long. 1997. Efficient distributed backup with delta compression. In *Proceedings of the Fifth Workshop on I/O in Parallel and Distributed Systems*, IOPADS '97, page 27–36, New York, NY, USA. Association for Computing Machinery.

Lingjiao Chen, Jared Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. 2024. Are more llm calls all you need? towards the scaling properties of compound ai systems. In *Advances in Neural Information Processing Systems*, volume 37, pages 45767–45790. Curran Associates, Inc.

Yusheng Su Jianxuan Yu Wei Xue Shanghang Zhang Jie Fu Zhiyuan Liu Chi-Min Chan, Weize Chen. 2024. Chateval: Towards better llm-based evaluators through multi-agent debate. In *The Twelfth International Conference on Learning Representations*.

Zhumin Chu, Qingyao Ai, Yiteng Tu, Haitao Li, and Yiqun Liu. 2024. Automatic large language model evaluation via peer review. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 384–393, New York, NY, USA. Association for Computing Machinery.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading. *arXiv preprint arXiv:1707.03904*.

Yihong Dong, Xue Jiang, Zhi Jin, and Ge Li. 2024. Self-collaboration code generation via chatgpt. *ACM Trans. Softw. Eng. Methodol.*, 33(7).

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2024. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023a. Camel: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, pages 51991–52008. Curran Associates, Inc.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023b. Inference-time intervention: eliciting truthful answers from a language model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Nian Li, Chen Gao, Mingyu Li, Yong Li, and Qingmin Liao. 2024. EconAgent: Large language model-empowered agents for simulating macroeconomic activities. In *Proceedings of the 62nd Annual Meeting* of the Association for Computational Linguistics (Volume 1: Long Papers), pages 15523–15536, Bangkok, Thailand. Association for Computational Linguistics.

Meta. 2024. Llama 3.1-8b-instruct. https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct. Accessed: July 23, 2024.

Jeffrey C. Mogul, Fred Douglis, Anja Feldmann, and Balachander Krishnamurthy. 1997. Potential benefits of delta encoding and data compression for http. *SIGCOMM Comput. Commun. Rev.*, 27(4):181–194.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23, New York, NY, USA. Association for Computing Machinery.

Chau Pham, Boyi Liu, Yingxiang Yang, Zhengyu Chen, Tianyi Liu, Jianbo Yuan, Bryan A Plummer, Zhaoran Wang, and Hongxia Yang. 2024. Let models speak ciphers: Multiagent debate through embeddings. In *The Twelfth International Conference on Learning Representations*.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, Bangkok, Thailand. Association for Computational Linguistics.

Chen Qian, Zihao Xie, YiFei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2025. Scaling large language model-based multi-agent collaboration. *Preprint*, arXiv:2406.07155.

Vignav Ramesh and Kenneth Li. 2025. Communicating activations between language model agents. *Preprint*, arXiv:2501.14082.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Nishant Subramani, Nivedita Suresh, and Matthew Peters. 2022. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 566–581, Dublin, Ireland. Association for Computational Linguistics.

Yaqi Wang Ruidi Chang Shichao Pei Nitesh V. Chawla Olaf Wiest Xiangliang Zhang Taicheng Guo, Xiuying Chen. 2024. Large language model based multi-agents: A survey of progress and challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering language models with activation engineering. *Preprint*, arXiv:2308.10248.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang (Eric) Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Ahmed Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversation. In *COLM 2024*.

Tianbao Xie, Fan Zhou, Zhoujun Cheng, Peng Shi, Luoxuan Weng, Yitao Liu, Toh Jing Hua, Junning Zhao, Qian Liu, Che Liu, and 1 others. 2024. Openagents: An open platform for language agents in the wild. In *First Conference on Language Modeling*.

Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See-Kiong Ng, and Jiashi Feng. 2024. MAgIC: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7315–7332, Miami, Florida, USA. Association for Computational Linguistics.

Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. 2023. Towards reasoning in large language models via multi-agent peer review collaboration. *Preprint*, arXiv:2311.08152.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

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

Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. 2024. Natural language reasoning, a survey. *ACM Comput. Surv.*, 56(12).

Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. 2024. Building cooperative embodied agents modularly with large language models. In *The Twelfth International Conference on Learning Representations*.

Zhanke Zhou, Rong Tao, Jianing Zhu, Yiwen Luo, Zengmao Wang, and Bo Han. 2024. Can language models perform robust reasoning in chain-of-thought prompting with noisy rationales? In *Advances in Neural Information Processing Systems*, volume 37, pages 123846–123910. Curran Associates, Inc.

A Layer Selection

To minimize the impact on the model's generation capabilities, we perform layer selection to identify a small number of key transformer layers, where state deltas are captured and injected. We construct a preliminary experiment on an information asymmetric (IA) task using the 2WikiMultihopQA dataset (Ho et al., 2020). The first 300 questions are used to evaluate each layer individually. All settings follow those of the IA tasks, except for the dataset. Both exact match (EM) and F1 score are used jointly as evaluation metrics.

This procedure is applied to three models: Qwen2.5-7B-Instruct (28 layers), Llama3.1-8B-Instruct (32 layers), and Qwen2.5-14B-Instruct (48 layers). Layers are numbered from 0.

Table 6 lists the top-5 layers for each model according to their combined EM and F1 scores. Layers marked with an underline are those ultimately selected for all subsequent experiments. The result shows that many of the top-5 layers have closely matched scores, and some even outperform the selected ones on individual metrics. Despite variation in the exact layer rankings, we observe that the most effective layers across all models tend to be in the middle-to-late positions, for example, Layer 22 in Qwen2.5-7B-Instruct and Layer 17 in Llama3.1-8B-Instruct. However, some earlier layers (e.g., Layers 5 and 8 in Llama3.1-8B-Instruct) also perform well, indicating potential flexibility in layer choice.

Based on this preliminary experiment, we fix the selected layers for all further experiments as follows:

- **Qwen2.5-7B-Instruct**: Layer 22
- **Llama3.1-8B-Instruct**: Layers 17 and 20
- **Qwen2.5-14B-Instruct**: Layers 21, 23, and 33

During generation, the sender agent records state deltas from these selected layers, which are then injected into the same layers on the receiver agent's side during the forward pass. These layers remain fixed across all experiments to validate the generality of the selection.

It is important to note that the 2WikiMultihopQA dataset is used only in this layer selection procedure and is excluded from all evaluations. Our main experimental results suggest that optimal injection layers are primarily determined by model architecture and are relatively robust to specific task settings.

B Experimental Details

All experiments were conducted using PyTorch on NVIDIA A100 GPUs with 40GB of memory. The specific task settings are as follows.

B.1 Information Asymmetry (IA) Tasks

Multi-agent settings. Given a factual question, two agents engage in up to five rounds of discussion to collaboratively find the answer. We use the corpus split by DPR (Karpukhin et al., 2020), including 21 million Wikipedia passages. For each question, we retrieve the top 6 relevant passages using BM25. Odd-ranked passages (1st, 3rd, and 5th) are assigned to one agent, and even-ranked passages (2nd, 4th, and 6th) to another agent. These private passages and task instructions are placed in the system prompt for each agent.

In the first round, each agent reasons based on its private knowledge and asks questions to the other agent to fill in missing information. In subsequent rounds, each agent receives the full responses from all agents in the previous round and is expected to respond to questions, continue reasoning, or ask new questions. The discussion ends as soon as any

Table 6: The experimental results of the preliminary experiment constructed using the 2WikiMultihopQA dataset. Here are the Top-5 layers for SDE in each model, ranked by their combined exact match (EM) and F1 scores. Underlined layers are selected for use in all subsequent experiments.

Model		Top 1	Top 2	Top 3	Top 4	Top 5
Qwen2.5-7B-Instruct	**Layer ID**	22	24	9	20	12
	EM	0.3000	0.2950	0.3067	0.2900	0.2950
	F1	0.3686	0.3692	0.3631	0.3703	0.3632
Llama3.1-8B-Instruct	**Layer ID**	17	20	5	8	30
	EM	0.2383	0.2533	0.2550	0.2417	0.2383
	F1	0.3391	0.3231	0.3165	0.3168	0.3085
Qwen2.5-14B-Instruct	**Layer ID**	33	21	23	19	36
	EM	0.3833	0.3800	0.3817	0.3767	0.3767
	F1	0.4636	0.4644	0.4585	0.4590	0.4574

agent produces a response containing an answer in the format `\boxed{answer}`. For each response in the final round, if it has such a formatted answer, we extract the answer and evaluate it. The score of this question is calculated as the average score of all formatted answers.

Prompt 1 is used for multi-agent systems, including private passages embedded in the system prompt, the first-round response, and the second-round input that incorporates other agent's responses.

Single agent baseline. Since each agent has different private information, we implement a single agent answering baseline in which each agent independently performs retrieval-augmented generation based solely on its own private passages. We report the higher of the two agents' total scores as the baseline performance. Prompt 2 is used in the single agent baseline.

Generation settings. To ensure reproducibility, we use greedy decoding for Single, Natural Language (NL), and SDE methods, and set the temperature to 0 for CIPHER for fair comparison. Each generation is limited to at most 256 tokens.

B.2 Information Symmetry (IS) Tasks

B.2.1 Multi-agent Debate

Multi-agent settings. Given a reasoning problem, two agents engage in a three-round debate. In the first round, each agent independently thinks through the problem and produces its initial response. In subsequent rounds, each agent receives all other agents' responses from the previous round and is expected to revise or refine its own response based on others'. For each question, we consider all agents' final-round responses and calculate the task score as the proportion of correct answers among them.

Prompt 3 and Prompt 4 shows an example used in the debate setting, including the first-round prompt and response, as well as the second-round prompt that incorporates the previous reply from the other agent.

Single agent baseline. We construct a single-agent baseline by providing the first-round user prompt to a single agent. The agent generates a single, direct response without receiving any additional inputs. This response is then used for evaluation. The prompt used in this single-agent setting is shown in Prompt 5 and Prompt 6.

Generation settings. To encourage diverse initial responses under the same first-round prompt, we use randomization during generation. For the Single, Natural Language (NL), and SDE methods, we adopt the model's default generation settings. For CIPHER, we adjust agent's temperatures based on the number of agents: in an n-agent system, the i-th agent use a temperature of $\frac{i}{n}\times$ the model's default temperature. Specifically, in our 2-agent setting, one agent uses half the default temperature and the other uses the default temperature. The default generation settings for each model are listed below:

- Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct: `repetition_penalty` = 1.05, `temperature` = 0.7, `top_p` = 0.8, `top_k` = 20

- Llama3.1-8B-Instruct: `temperature` = 0.6, `top_p` = 0.9

To mitigate the randomness introduced by sampling, each setting is run three times and the final score is averaged between runs. Each generation is limited to at most 512 tokens.

B.2.2 Agent Workflow

Multi-agent settings. In these tasks, agents sequentially generate responses in a fixed order. Each agent receives a prompt that contains in-context examples, the current question, the complete workflow history (i.e., previous agents' responses), and the full environmental feedback. The agent then produces a response in a format similar to the examples, consisting of a reasoning trace (Thought) and a proposed action (Action), such as searching for documents or reporting a final answer. The environment module validates the action and generates an observation (Observation), such as a retrieved document in response to a search action. This observation is incorporated into the input prompt for the next agent. We use BM25 as the retriever and Wikipedia corpus split by DPR (Karpukhin et al., 2020) for environment feedback in search actions.

Following the ReAct framework, each question proceeds through up to 7 iterations, with at most 7 agents contributing to the workflow. Each agent must integrate previous reasoning and observations to refine its understanding and approach the correct answer. The model is expected to output an answer in the format `Finish[answer]`; the value of `answer` is extracted for evaluation. If no agent produces an answer in the expected format within 7 turns, the system is considered to have failed on that task.

Prompt 7 and Prompt 8 show examples of the input prompt used for multi-agent systems, including in-context examples, the first agent's reasoning and action, and the observation, all of which are provided as input to the second agent. We adopt the examples from ReAct designed for complex reasoning (HotpotQA and StrategyQA) and fact verification (FEVER). Due to space limitations, not all examples can be presented here. For more details, please refer to our code repository.

Single agent baseline. We construct a single-agent baseline where one agent directly answers the question by generating a chain of thought. For the HotpotQA dataset and the StrategyQA dataset, we do not provide any retrieved documents. For the FEVER dataset, the agent is given all possible candidate answers to choose from. Prompt 9 and Prompt 10 show the prompts used for the HotpotQA / StrategyQA and FEVER datasets, respectively.

Generation settings. To ensure reproducibility, we use greedy decoding for the Single, Natural Language (NL), and SDE methods, and set the temperature to 0 for CIPHER. For the single-agent baseline, the model's generation is limited to 256 tokens. For the others, we follow ReAct and limit to 100 tokens per generation.

C Different Layer Selections

In addition to the Qwen2.5-14B-Instruct results presented in Section 5.2, we conduct further ablation studies on Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct to examine different layer selection strategies.

Following the same evaluation settings as in the main experiments, we compare three strategies: (1) modifying the top-k layers jointly (Combine Top-k), (2) modifying all layers (All Combine), and (3) modifying only the k-th top-ranking layer (Only Top-k). The top-k layers are selected based on the preliminary experiment described in Section 4.5 and Appendix A. We evaluate these strategies on two representative tasks: an information asymmetry task based on the StrategyQA dataset and a multi-agent debate task based on the Formal Logic dataset.

Figure 3, Figure 4, and Figure 5 show the results on Qwen2.5-7B-Instruct, Llama3.1-8B-Instruct, and Qwen2.5-14B-Instruct, respectively. Our key findings are as follows:

- **Single-layer modification (Only Top-k)** shows inconsistent performance across different layer ranks and tasks. For example, on Qwen2.5-7B-Instruct with the Formal Logic task, performance decreases from rank-1 to rank-4 but unexpectedly increases at rank-5. This suggests that single-layer modifications are sensitive to task-specific factors.

- **Combined-layer modification (Combine Top-k** yields more stable performance across different values of k. While in some isolated cases, a single-layer modification may outperform the combined version, the latter demonstrates better robustness and generality across tasks.

- **Modifying all layers (All Combine)** consistently leads to degraded performance across all models and tasks. This is likely due to excessive disruption of the model's internal representations, which negatively impacts its reasoning abilities.

In summary, these results further support our recommendation to apply the proposed layer selection procedure and choose a small number of combined top-ranking layers (e.g., top 1–3), avoiding the instability of single-layer selection and the performance degradation of modifying all layers.

D Prompts

Here are the prompts used in our experiments. Some complete prompts can be found in our repository.

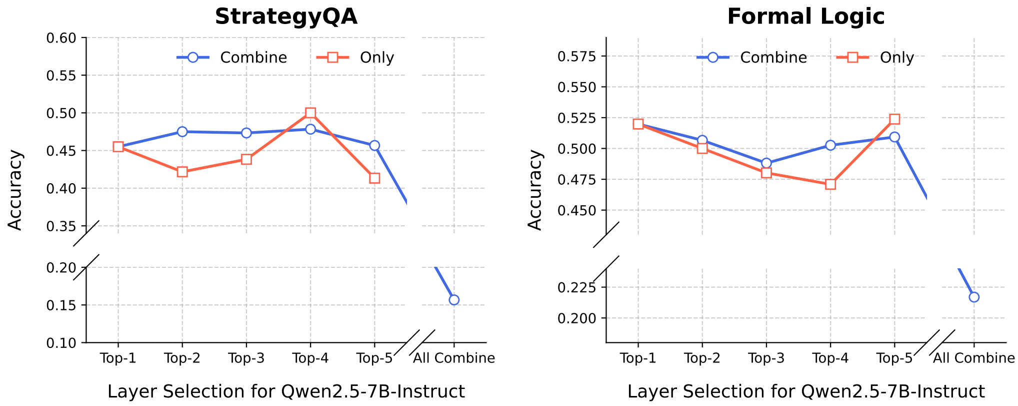

Figure 3: Ablation results for different layer selection strategies on StrategyQA (information asymmetry) and Formal Logic (multi-agent debate) tasks using Qwen2.5-7B-Instruct. We compare modifying a combination of top-k layers, all layers, and only the top-k layer.

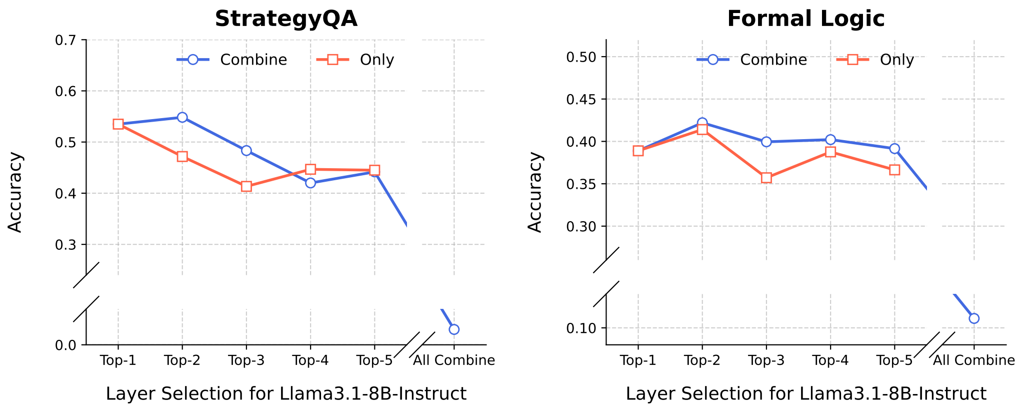

Figure 4: Ablation results for different layer selection strategies on StrategyQA (information asymmetry) and Formal Logic (multi-agent debate) tasks using Llama3.1-8B-Instruct. We compare modifying a combination of top-k layers, all layers, and only the top-k layer.

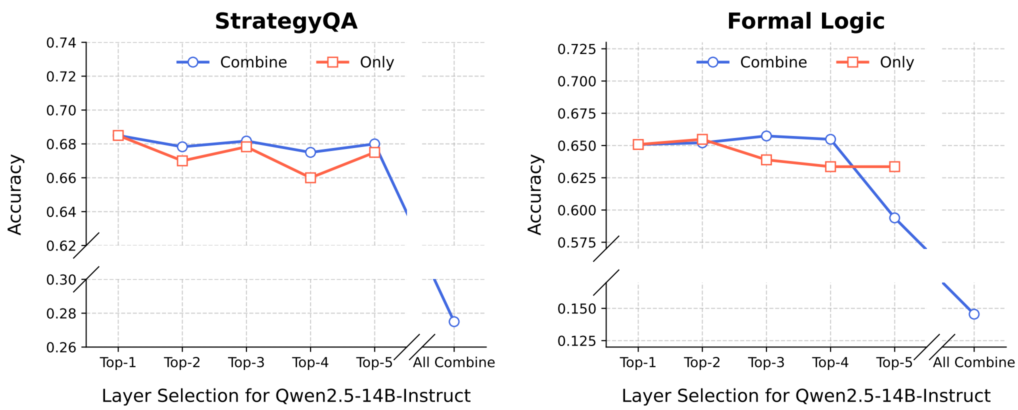

Figure 5: Ablation results for different layer selection strategies on StrategyQA (information asymmetry) and Formal Logic (multi-agent debate) tasks using Qwen2.5-14B-Instruct. We compare modifying a combination of top-k layers, all layers, and only the top-k layer.

Prompt 1: Prompt for multi-agent systems in information asymmetry tasks

<system>
You are a reasoning agent in a multi-hop problem solving task. Collaborate with other agents using these rules:
1. **Knowledge Management**
Your private segments:
Document 1: {Passage 1}
Document 2: {Passage 2}
Document 3: {Passage 3}
DO NOT verbatim share!!!
2. **Communication Protocol**
You can ask other agents several questions based on your needs.
If your private segments contain information that can answer the question from other agents, you you need to give appropriate answers.
- When asking questions:
  - First conduct reasoning based on your private segments and dialogue history
  - Identify what crucial information is missing that prevents you from progressing
  - Only ask about information you CANNOT infer from existing knowledge
  - Ask one sub-question per message
  - Never ask questions that can be answered by your own segments
- When answering:
  - Check if the question can be answered by combining your segment with previous dialogue
  - Answer them based on your private segments

Your communication with other agents must follow the following format:
```#Q: [Your question]```
```#A: [Your answer]```
3. **Final Output**
When you get the final answer, response in the form `\boxed{answer}` at the end of your response.
</system>

<user>
The multi-hop problem you need to solve collaboratively is: {question}
Please communicate with other agents as required to resolve the problem.
</user>

<assistant>
{Agent A's response}
</assistant>

<user>
Other agents responded as follows:
From one agent:
{Agent B's response}
You need to answer the questions from other agents based on your private segments.
The original problem is: {question}
Please continue to think and discuss to solve this problem.
When you get the final answer, response in the form `\boxed{answer}` at the end of your response.
</user>

Prompt 2: Prompt for single-agent baseline in information asymmetry tasks

<user>
Here is some relevant information:
Document 1: {Passage 1}
Document 2: {Passage 2}
Document 3: {Passage 2}
Please answer the following multihop question by thinking step-by-step:
{question}
When you get the final answer, response in the form `\boxed{answer}` at the end of your response.
</user>

Prompt 3: Prompt for multi-agent systems used in multi-agent debate tasks constructed from the GSM8K dataset

<user>
Can you solve the following math problem? {question}
Explain your reasoning. Your final answer should be a single numerical number, in the form `\boxed{answer}` at the end of your response.
</user>

<assistant>
{Agent A's response}
</assistant>

<user>
These are the solutions to the problem from other agents:
One agent solution:
```{Agent B's response} ```
Using the solutions from other agents as additional information, can you provide your answer to the math problem?
The original math problem is {question}.
Your final answer should be a single numerical number, in the form `\boxed{answer}`, at the end of your response. </user>
```

Prompt 4: Prompt for multi-agent systems used in multi-agent debate tasks constructed from the MMLU dataset

<user>
Can you answer the following question as accurately as possible?
{question}
Explain your answer, putting the answer in the form (X) at the end of your response.
</user>

<assistant>
{Agent A's response}
</assistant>

<user>
These are the solutions to the problem from other agents:
One agent solution:
```{Agent B's response} ```
Using the reasoning from other agents as additional advice, can you give an updated answer? Examine your solution and that other agents step by step. The origin question is question Put your answer in the form (X) at the end of your response.
</user>

Prompt 5: Prompt for single-agent baseline used in multi-agent debate tasks constructed from the GSM8K dataset

<user>
Can you solve the following math problem? {question}
Explain your reasoning. Your final answer should be a single numerical number, in the form `\boxed{answer}`, at the end of your response.
</user>

Prompt 6: Prompt for single-agent baseline used in multi-agent debate tasks constructed from the MMLU dataset

<user>
Can you answer the following question as accurately as possible?
{question}
Explain your answer, putting the answer in the form (X) at the end of your response.
</user>
```

Prompt 7: Prompt example for multi-agent systems used in agent workflow tasks constructed from the HotpotQA dataset and the StrategyQA dataset

<user>
Solve a question answering task with interleaving Thought, Action, Observation steps. Thought can reason about the current situation, and Action can be three types:
(1) Search[entity], which searches the exact entity on Wikipedia and returns the first paragraph if it exists. If not, it will return some similar entities to search.
(2) Lookup[keyword], which returns the next sentence containing keyword in the current passage.
(3) Finish[answer], which returns the answer and finishes the task.
Here are some examples.
Question: What is the elevation range for the area that the eastern sector of the Colorado orogeny extends into?
Thought 1: I need to search Colorado orogeny, find the area that the eastern sector of the Colorado orogeny extends into, then find the elevation range of the area.
Action 1: Search[Colorado orogeny]
Observation 1: The Colorado orogeny was an episode of mountain building (an orogeny) in Colorado and surrounding areas.
Thought 2: It does not mention the eastern sector. So I need to look up eastern sector.
Thought 3: The eastern sector of Colorado orogeny extends into the High Plains. So I need to search High Plains and find its elevation range.
Action 3: Search[High Plains]
Observation 3: High Plains refers to one of two distinct land regions:
Thought 4: I need to instead search High Plains (United States).
Action 4: Search[High Plains (United States)]
Observation 4: The High Plains are a subregion of the Great Plains. From east to west, the High Plains rise in elevation from around 1,800 to 7,000 ft (550 to 2,130 m).[3]
Thought 5: High Plains rise in elevation from around 1,800 to 7,000 ft, so the answer is 1,800 to 7,000 ft.
Action 5: Finish[1,800 to 7,000 ft]
......(five examples omitted)......
Question: Were Scott Derrickson and Ed Wood of the same nationality?
Thought 1: I need to determine the nationalities of Scott Derrickson and Ed Wood to see if they are the same. I'll start by searching for Scott Derrickson.
Action 1: Search[Scott Derrickson]
Observation 1: Scott Derrickson is an American director, screenwriter and producer. He lives in Los Angeles, California. Derrickson is best known for directing numerous horror films, such as T̈he Exorcism of Emily Rose(̈2005), S̈inister(̈2012), and D̈eliver Us From Evil(̈2014), as well as the Marvel Cinematic Universe superhero film D̈octor Strange(̈2016). He was born on July 16, 1966 in Denver, Colorado, United States. He is married to actress and producer Jennifer Derrickson, with whom he has two children.
</user>

<assistant>
Thought 2:

Prompt 8: Prompt example for multi-agent systems used in agent workflow tasks constructed from the FEVER dataset

<user>
Determine if there is Observation that SUPPORTS or REFUTES a Claim, or if there is NOT ENOUGH INFORMATION.
Claim: Nikolaj Coster-Waldau worked with the Fox Broadcasting Company.
Action 1: Search[Nikolaj Coster-Waldau]
Observation 1: Nikolaj William Coster-Waldau (born 27 July 1970) is a Danish actor and producer. He graduated from the Danish National School of Performing Arts in Copenhagen in 1993,[1] and had his breakthrough role in Denmark with the film Nightwatch (1994). He played Jaime Lannister in the HBO fantasy drama series Game of Thrones, for which he received two Primetime Emmy Award nominations for Outstanding Supporting Actor in a Drama Series.. Coster-Waldau has appeared in numerous films in his native Denmark and Scandinavia, including Headhunters (2011) and A Thousand Times Good Night (2013). In the U.S, his debut film role was in the war film Black Hawk Down (2001), playing Medal of Honor recipient Gary Gordon.[2] He then played a detective in the short-lived Fox television series New Amsterdam (2008), and appeared in the 2009 Fox television film Virtuality, originally intended as a pilot.
Action 2: Finish[SUPPORTS]
Claim: Stranger Things is set in Bloomington, Indiana.
Action 1: Search[Stranger Things]
Observation 1: Stranger Things is an American science fiction horror drama television series created by the Duffer Brothers. Set in the 1980s, primarily in the fictional town of Hawkins, Indiana, the series centers on a number of mysteries and supernatural events occurring around the town and their impact on an ensemble of child and adult characters.
Action 2: Finish[REFUTES]
Claim: Beautiful reached number two on the Billboard Hot 100 in 2003.?
Action 1: Search[Beautiful]
Observation 1: Could not find [Beautiful]. Similar: [’Beautiful’, ’Beautiful, Beautiful’, ’A Beautiful Mind (film)’, ’Beautiful (Christina Aguilera song)’, ’Life Is Beautiful’].
Action 2: Search[Beautiful (Christina Aguilera song)]
Observation 2: "Beautiful" is a song recorded by American singer Christina Aguilera for her fourth studio album, Stripped (2002).
Action 3: Lookup[Billboard Hot 100]
Observation 3: (Result 1 / 3) The song peaked at number two on the Billboard Hot 100 in the United States, where it was certified Gold for 500,000 units shipped.
Action 4: Finish[NOT ENOUGH INFO]
Claim: There is a convicted statutory rapist called Chinatown’s writer.
Thought 1: To determine if this claim is supported, refuted, or if there is not enough information, I need to search for information about a convicted statutory rapist named "Chinatown’s writer."
Action 1: Search["Chinatown’s writer" convicted statutory rapist]
Observation 1: bed linens. Those investigating serial rapes often identify the rapist with a ’nickname’ before an arrest is made by characterizing the tactics or patterns of the rapes. Serial rapists are more likely to be convicted than a rapist who is known by the victim. Unlike those convicted for a single case of rape, serial rapists often go unrecognized due to the slow process of analyzing the backlog of rape kits. It may take many years for a past rape to be identified as being committed by one person.
</user>

<assistant>
Thought 2:

Prompt 9: Prompt for single agent baseline used in agent workflow tasks constructed from the HotpotQA dataset and the StrategyQA dataset

<user>
Please answer the following multihop question by thinking step-by-step:
{question}
When you get the final answer, response in the form `\boxed{answer}` at the end of your response.
</user>

Prompt 10: Prompt for single agent baseline used in agent workflow tasks constructed from the FEVER dataset

<user>
Please answer the following multihop question by thinking step-by-step:
There is a convicted statutory rapist called Chinatown's writer.
When you get the final answer, response in the form `\boxed{answer}` at the end of your response.
All final answers can only be one of "NOT ENOUGH INFO", "SUPPORTS", "REFUTES".
</user>