CMAT: A Multi-Agent Collaboration Tuning Framework

Accepted by ICME’25

ICLR 2025 Workshop

Presented: Yinghui XIA (HKUST-GZ)

AutoAgents
未 来 式 智 能

Presentation Outline

AutoAgents

1.Introduction

- **CMAT :** Collaborative Multi-Agent Tuning Framework
- **Problem:** LLMs still heavily rely on human input for dialogue guidance
- **Solution:** TinyAgent model + CMAT framework
- **Key Innovation:** Autonomous agents that can steer conversations with minimal human supervision
- **Significance:** Small models (TinyAgent-7B) achieving performance comparable to larger models (GPT-3.5)

2. Background and Motivation

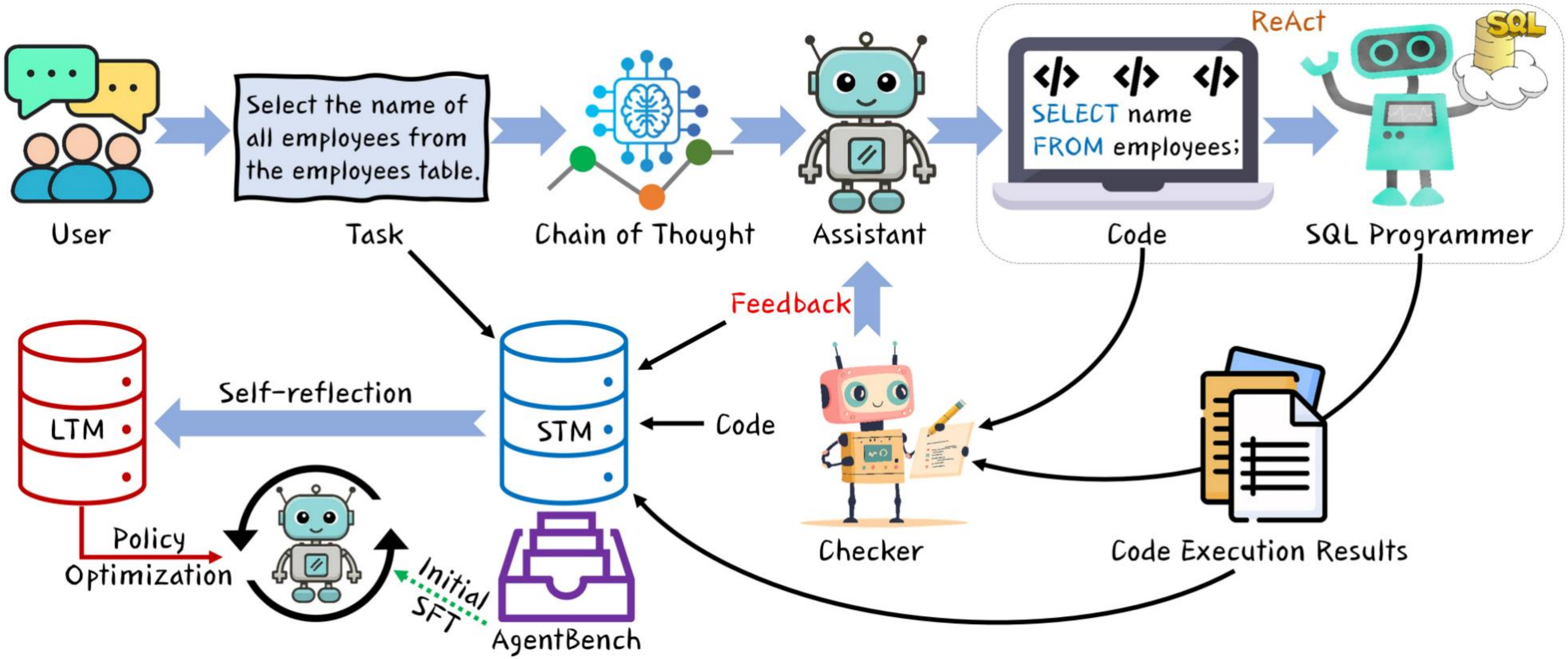
Current LLM Challenges

- High computational requirements
- Data biases and lack of robustness
- Limited applicability in resource-constrained environments
- Heavy reliance on human guidance

Research Question

"Can we replace human intervention with autonomous autonomous communication agents capable of steering steering conversations towards task completion with with minimal human supervision?"

3. The CMAT Framework



3. The CMAT Framework

Key Components	
Structured Environment:	Individual agents with specialized roles
Collaborative Decision-Making:	Agents work together to process information and solve tasks
Adaptive Learning:	Real-time adaptation through environmental feedback
Role-Playing Mechanism:	Precise task allocation and enhanced agent communication

4. TinyAgent Models

- TinyAgent-1.8B

Fine-tuned version of Qwen-1.8B

- TinyAgent-7B

Fine-tuned version of CodeLlama-7B

- Performance

Rivals models with significantly more parameters

AutoAgents 未来式智能

5. Methodology

Agent Roles

- User (U):

Provides input tasks and instructions

- Assistant (A):

Acts as the "Actor," generating actions based on policy

- Checker (C):

Serves as the "Critic," evaluating Assistant's actions and providing providing feedback

Learning Strategy

- Supervised Fine-Tuning:

Using LoRA and P-Tuning techniques

- Chain of Thought (CoT):

Generating intermediate reasoning steps

- Feedback-Driven Policy Optimization:

Actor-Critic inspired approach

- Memory Management:

Dual-memory system (short-term and long-term)

5. Methodology

Supervised fine-tuning formula:

$$L_{\text{sup}}(\theta_{\text{actor}}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[\ell\left(\mathcal{M}_{\theta_{\text{actor}}}(\mathbf{x}), \mathbf{y}\right)\right]$$

Feedback based strategy optimization formula

A strategy update formula inspired by Actor Critic dynamics:

$$\theta_{\text{actor}} \leftarrow \theta_{\text{actor}} + \alpha \nabla_{\theta_{\text{actor}}} \log \pi_{\theta_{\text{actor}}}(\mathbf{a}_t | s_t)\delta_t$$

Feedback based strategy optimization formula:

$$\theta_{\text{critic}} \leftarrow \theta_{\text{critic}} + \beta \delta_t \nabla_{\theta_{\text{critic}}} V_{\theta_{\text{critic}}}(s_t)$$

Formula for updating strategy parameters based on feedback and self reflection:

$$\theta_{\text{actor}} \leftarrow \theta_{\text{actor}} - \alpha \nabla_{\theta_{\text{actor}}} L(f_t, \mathbf{a}_t) + \gamma \nabla_{\theta_{\text{actor}}} G(s_t)$$

6. Experiments

Evaluation Framework

- Six key domains tested
- Operating Systems (OS)
- Database Management (DB)
- Knowledge Graphs (KG)
- ALFWorld (ALF)
- WebShop (WS)
- Mind2Web (M2W)

Performance Comparison

- TinyAgent models vs. API-based models (GPT-3.5, GPT-4)
- TinyAgent models vs. other open-source models
- Impact of CMAT framework on overall performance

7. Key Experimental Results

Table 2: Test set results of AGENTBENCH. Comparison between API-based models and open-source models. Bold: The best among API-based and open-source models.

LLM Type	Models	VER	OS	DB	KG	ALF	WS	M2W
API	gpt-3.5-turbo	613	31.6	15.7	25.9	16.0	**64.1**	16.0
	gpt-4	613	**42.4**	**32.0**	**58.8**	**78.0**	61.6	**29.0**
	text-davinci-003	–	20.1	16.3	34.9	20.0	61.7	26.0
	text-davinci-002	–	8.3	16.7	41.5	16.0	56.3	9.0
OSS	tinyllama-1.1b[1]	–	2.8	0.0	0.0	0.0	0.0	0.0
	opt-1.3b[2]	–	0.7	0.0	0.0	0.0	0.0	0.0
	opt-2.7b	–	1.4	0.0	0.0	0.0	0.0	0.0
	qwen-1.8b	chat	10.4	22.67	6.8	0.0	26.6	5.0
	chatglm2-6b[3]	v1.1	4.2	1.3	0.0	0.0	0.0	0.0
	codellama-7b	instruct	9.7	2.7	0.0	0.0	14.3	5.0
	llama2-7b[4]	chat	0.0	4.2	8.0	0.0	11.6	7.0
	zephyr-7b[5]	alpha	12.5	9.7	5.0	8.0	45.0	11.0
	baichuan2-6b[6]	chat	2.8	9.7	0.0	0.0	6.1	11.0
	mpt-7b[7]	chat	5.6	9.7	12.7	0.0	0.0	0.0
	qwen-7b	chat	12.5	13.0	7.0	**34.3**	0.0	0.0
	agentlm-7b	chat	14.6	33.0	9.0	16.4	18.4	10.0
	agentlm-7b(SFT)	chat	17.4	37.0	10.0	17.4	26.6	10.0
	tinyagent-1.8b	chat	17.7	28.33	**48.0**	6.0	32.7	11.0
	tinyagent-7b	chat	**23.1**	**41.3**	28.0	8.0	**58.7**	**12.0**

7. Key Experimental Results

Table 1: Evaluation of Code Correction

Model	BLEU-4	ROUGE-1	ROUGE-2	ROUGE-L
codellama-7b	25.01	45.91	29.83	26.24
codellama-13b	26.96	45.31	29.54	25.91
tinyllama-1.8b	**43.38**	**59.86**	**37.81**	**42.86**

The superior performance of TinyLlama-1.8b in code correction tasks compared to larger models.

Model size isn't the only determinant of performance - efficient training and quality datasets also play crucial roles in crucial roles in model effectiveness.

8. Ablation Study and Analysis

Table 5: Ablation study on the effect of agent and general instructions.

Models	OS	DB	KG	ALF	WS	M2W
TinyAgent-7B	**27.3**	**43.0**	**38.0**	**10.0**	**61.8**	**14.0**
– Agent only	20.1	39.3	25.0	2.0	55.7	7.0
– General only	9.7	5.4	0.0	0.0	26.6	5.0

The removal of any component will affect the effectiveness of CMAT

This confirms that the integration of complementary instruction types enables more more sophisticated capabilities than either component alone.

9. Limitations

- Model Variability

Framework effectiveness varies between models

- Dataset Constraints

Limited by quality and diversity of training data

- Task Type Limitations

Performance varies across different task types

- Sim to Real Limitation

Evaluation may not fully reflect complex real-world scenarios

- Computational Resources

Larger-scale models couldn't be tested due to resource constraints

10. Conclusions

Main Contributions

- CMAT framework allowing dynamic and real-real-time memory updates
- Novel role-playing mechanism for task allocation and agent communication
- Fine-tuned TinyAgent models competing with competing with advanced LLMs

Key Findings

- Small-parameter models with quality training data can match larger models
- CMAT framework significantly enhances model performance
- Importance of prompt design and quality in quality in optimizing model performance performance

Future Directions

- Wider range of models and datasets
- Implementation in resource-constrained constrained environments
- More complex real-world applications

Thank You

Autoagents.ai
corresponding: edward.yang@autoagents.ai