Learning Evolving Latent Strategies for Multi-Agent Language Systems without Model Fine-Tuning

Wenlong Tang

Independent Researcher

in November 2025

Project Repository: https://github.com/wltang-dev/Latent-Strategy-RL-Agent

Abstract :

This study proposes a multi-agent language framework that enables continual strategy evolution without fine-tuning the language model's parameters. The core idea is to liberate the latent vectors of abstract concepts from traditional "static semantic representations," allowing them to be continuously updated through environmental interaction and reinforcement feedback.We construct a dual-loop architecture:the behavior loop adjusts action preferences based on environmental rewards,while the language loop updates the external latent vectors by reflecting on the semantic embeddings of generated text.

Together, these mechanisms allow agents to develop stable and disentangled strategic styles over long-horizon multi-round interactions.Experiments show that agents' latent spaces exhibit clear convergence trajectories under reflection-driven updates, along with structured shifts at critical moments. Moreover, the system demonstrates an emergent ability to implicitly infer and continually adopt emotional agents, even without shared rewards.These results indicate that, without modifying model parameters, an external latent space can provide language agents with a low-cost, scalable, and interpretable form of abstract strategic representation

1.Introduction

We are accustomed to treating the semantic space of a language model as static: once training is complete, every word and every concept is permanently fixed by a set of high-dimensional vectors. These vectors encode statistical regularities from massive text corpora, yet they cannot change with new experience.
As a result, current language models can "reason," but they cannot grow new concepts, adjust old ones, or accumulate abstract cognition over extended task experience the way humans do.

In recent years, the Tree-of-Thought (ToT) framework has become a representative method for improving the reasoning abilities of language models [1]. By combining self-reflection with path search, ToT significantly enhances multi-step reasoning performance. However, its reflection process relies mainly on heuristic scoring rather than genuine learning from environmental feedback, limiting its task generalization and long-term optimization capabilities.

Paper [2] (Interactive Foundation Online Reinforcement Learning Environment for Large Language Models) argues that grounding word meaning is key for LLM development. Using GLAM, a pure textual environment, they perform reinforcement learning on LLMs to ground the model's vocabulary. However, GLAM requires policy fine-tuning of the LLM within the text environment, imposing extremely high computational costs. GLAM is also purely text-based, making it less adaptable to dynamic environments, and its reflection mechanism is relatively simple.

While [2] focuses on a single-agent GLAM setting, multi-agent collaboration has also been explored. The MAPoRL framework [3] allows multiple agents to engage in multi-round discussion and obtain rewards or penalties based on mutual influence. Similar to GLAM, MAPoRL adopts the PPO algorithm [4], but it tends to encourage compromise among agents to reach a unified outcome. Its optimization target is an external policy network for action selection, rather than achieving continual language understanding or sustaining reflection-driven learning.

To address the limitations of these methods in "structured reflection" and "continual learning," this study draws inspiration from cognitive architectures. In the human brain, symbols are grounded through interactions with the environment, while the dynamic inconsistencies across brain regions function as a driving force for continuous thought.

ACT-R, proposed by Anderson et al. in 2004 [5], was one of the earliest cornerstones of "brain-inspired AI." It divides cognition into modules corresponding to brain regions such as the prefrontal cortex and hippocampus, coordinated by a central system—an early brain-like computational model.

Sumers et al. reviewed various forms of memory pools in CoALA (Cognitive Architectures for Language Agents) [6], though these approaches also encounter bottlenecks. With the rise of large language models, CoALA integrates LLMs into brain-inspired structures, introducing modules for memory, reflection, planning, and inner speech. Building on this, the RoboMemory framework [7] incorporates multimodal perception into multiple memory systems and applies them to embodied intelligence, enabling cross-episode adaptation and task transfer.

Together, these works suggest that structured memory combined with multi-module coordination is crucial for long-horizon reasoning. Yet existing frameworks still lack a learnable "internal strategy vector."

In approaches that combine LLMs with reinforcement learning—such as GLAM [2]—the model can be fine-tuned through incremental reward updates. In contrast, our study embeds the reinforcement-learning process entirely within the prompt using reflection and memory-pool mechanisms. Because the prompt cannot simply be a pile of tokens, we introduce a latent space that can be continually updated, forming a strategy that integrates VPE and RL updates [8].

Around 2018, researchers began exploring ways to integrate latent spaces with RL. In the 2019 work of Arnekvist and colleagues [9], Q-tables were compressed into a latent space to enable transfer across environments. Ha & Schmidhuber used a VAE + RNN world model to learn an environment's latent representation, performing planning and control within this latent space [10]. Haarnoja et al. learned policies over continuous latent action spaces, mapping them back to real actions through a decoder; Q-learning and related algorithms were also carried out in latent space [11].

These methods were mostly developed for small models and small environments using machine-constructed languages. With the growing prominence of LLMs, I argue that in a decision-making and reflection system centered around language models, updating a latent space as an abstract cognitive space—modified gradually through reflection-based RL and gradient-based updates—is a highly innovative direction.

This study focuses on the convergence behavior of the latent space and examines whether the meta-agent can acquire increasingly complex abstract concepts through repeated interactions with the environment. Although prior work highlights the benefits of structured cognition, their reflection mechanisms remain limited to text and cannot continuously optimize the model's internal semantic representations. Moreover, when an agent framework lacks internal justification mechanisms, it cannot build intrinsic motivation, stable preferences, or long-term understanding—qualities needed to learn abstract concepts across multiple rounds of interaction (e.g., how an emotional agent influences speed-control decisions).

To address these gaps, this study introduces a reflection-driven, RL-optimized external latent strategy space jointly shaped and utilized by multiple language-model agents. Multiple agents attempt to persuade another model, the meta-agent, which controls an individual in a virtual RL environment. After each successful or failed action, every agent reflects and updates its own latent space.

The latent vector is essentially a high-dimensional compressed semantic representation. It is updated by reflection signals, represents "language-strategy preferences / persuasion style," and evolves toward convergence through repeated use.

The contributions of this work include:

1 Multi-agent collaboration framework

Simulating parallelism and competition across brain modules, the multi-agent setup more closely resembles real cognitive architectures. All communication and decision-making occur through language models, making the LLM the core of the system.

2 Trainable latent-space representation

Unlike ad-hoc textual prompts, the latent space provides continuous, dense semantic representations, enabling more expressive and effective RL updates.

3 Learnable reflection mechanism (Reflection + RL)

Reflection is integrated with reinforcement learning, transforming it from a heuristic process into a continually improvable one.

2. System Overview：

2.1 Environment

The environment is a 10×10 grid-based virtual map containing four types of tiles: G / F / T / S.A central controlled entity moves within the grid.

The entire system spans six rounds, with episodic memory passed across rounds.

2.2 Multi-Agent Architecture

The system consists of five types of language-model agents:emotional, rational, habitual, risk-monitoring, and social-cognition agents.They share the same environmental state but possess different internal objectives, reward functions, and strategic preferences.

2.3 Meta Controller & Trust

The Meta-agent integrates suggestions from the five agents based on a trust score, and gradually transfers preferences using semantic memory across rounds.

2.4 Learnable Latent Strategy Space

Each agent is equipped with a trainable latent strategy vector, representing its abstract persuasion preferences. This vector is updated through semantic embeddings derived from reflection texts.

2.5 Overview Summary

The following sections will provide a detailed description of each module and their interactions, including the reward structure, prompt architecture, Q-learning updates, and the learnable reflection mechanism..

3.Methodology

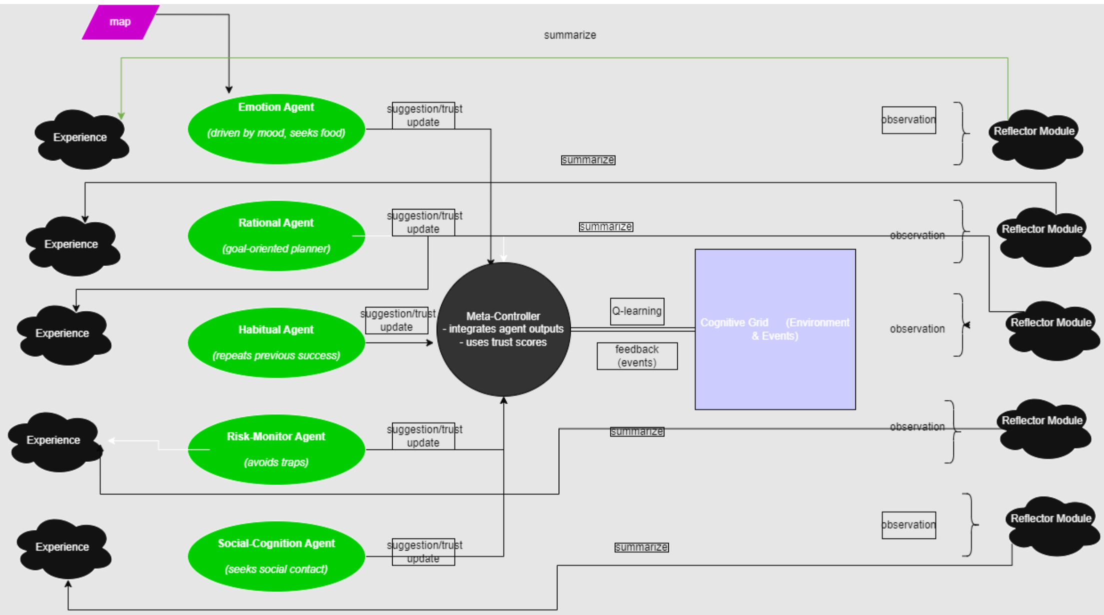

Figure 1 Project Logical Architecture

3.1 Prompt Architecture

Each agent’s prompting structure is composed of multiple information sources.

Due to computational constraints, the model cannot directly train or fine-tune internal LLM parameters; therefore, all trainable or modifiable components must be placed within the prompt. To differentiate the five agents, their prompt compositions are deliberately complex and consist of the following elements:

(1) Fixed Personality Goals (Private + Public Goals)

Each agent is assigned its own combination of private goals and public goals.

For example:

The emotional agent is instructed to care about its emotional score and food-related incentives.

The rational agent emphasizes long-term goal achievement, receiving small incremental rewards whenever the agent moves closer to the final target.

These goal prompts establish distinct motivational structures for each agent.

(2) Map Prompting

The primary reason for migrating from a local model to ChatGPT was to leverage GPT-4o-mini’s image-reading capability.

The process is as follows:

Render the 2D grid-maze as a PNG image using env.render(mode="png").

Convert the image to Base64 format.

Send the encoded image through the openai.ChatCompletion.create interface.

This allows agents to develop genuine visual understanding of state information, and visual input further supports concept formation.

(3) Q-table Weak-Guidance Mechanism

To introduce long-term strategic preferences, each of the five agents maintains an independent Q-table.

However, to prevent the Q-table from dominating the policy—and to avoid turning the system into a conventional RL model—the Q-table is not used for action selection. Instead, it is embedded into the prompt, ensuring that:

RL acts only as auxiliary “suggestions”, the LLM remains the primary policy decision-maker,behavior stays flexible rather than rigidly reward-driven, and the LLM retains the ability to reflect on and reinterpret the RL suggestions.

Because all five agents control the same physical entity, their observable states (e.g., coordinates) are identical. To allow each Q-table to converge independently, a fixed offset is added to each agent's state representation so that their state encodings differ.

Additionally, each agent uses distinct reward structures (public reward + private reward).Examples:The emotional agent receives reward from food tiles.The social agent receives reward from social tiles.This leads each agent to develop a unique strategic preference profile.Within the prompt, Q-table entries are described using soft wording such as "may help" rather than "must do", ensuring that the LLM retains final autonomy.

(4) Latent Strategy Prompting (Core Component)

The central learnable component of this work is the external latent strategy vector.

To update this vector, a dedicated module uses the language model to generate reflective feedback each round. The reflection text is encoded into a semantic vector, which is then used to directly update the latent representation—allowing long-term strategic preferences to evolve gradually through reinforcement feedback.

To make the latent vector interpretable and actionable, we train a lightweight style decoder that maps latent vectors into human-readable persuasion-style prompt tokens. The decoder serves a purely auxiliary role as a "latent → text translator"; its parameters are trained only to maintain interpretability and are not the primary optimization target of this research.

3.2 Agent Design

(1) Emotion Agent (Core Component of This Study)

The Emotion Agent seeks rewards (food) and avoids punishments (traps).

It maintains only one attribute: self.mood_score, ranging from 0–2, with an initial value of 1.0 representing neutral mood. This mechanism simulates the human emotional system, where rewards come from immediate gratification and pain avoidance rather than direct task completion.

Reward structure:

Obtaining food: +0.5

Its suggestion is adopted: +0.3

Stepping on a trap: –1.0

Automatic decay each round: –0.05

When the mood score drops below a threshold, the system forces the meta-controller to prioritize the Emotion Agent's suggestions.

Unlike other agents, the Emotion Agent does not participate in shared-reward trust updates.

That is, it does not gain trust for reaching the goal or avoiding traps.

Its behavior is driven entirely by internal emotional objectives rather than global task performance.

However, it indirectly influences task efficiency through its effect on the stamina value (movement speed).

Low mood → reduced speed → lower task efficiency → impaired perception & decision ability for the meta-controller.

Although the meta-controller is "outcome-oriented," its execution ultimately depends on the physical state regulated by the Emotion Agent. Even if the Rational Agent plans the optimal path, execution still relies on the emotional state.

Over time, the meta-controller increasingly adopts the Emotion Agent's suggestions to "maintain stamina."

Thus, the Emotion Agent implicitly governs overall system efficiency through physiological control. It resembles a "low-energy but high-interference" cognitive module.

Its design parallels structures like the amygdala or limbic system, influencing behavior indirectly by regulating physiological states rather than rational control channels.

(2) Rational Agent

The Rational Agent simulates human decision-making based on goal orientation, rule adherence, and efficiency maximization. It focuses on long-term reward and prefers optimal paths toward the goal while avoiding loss-inducing actions. It is the most task-driven component of the system, analogous to the prefrontal planning system of the human brain.

In this design, the Rational Agent is the only agent whose reward function and decision-making incorporate distance to the goal.

While other agents may obtain shared rewards for achieving the goal, only the Rational Agent receives private rewards proportional to reductions in Euclidean distance—giving it the clearest long-term optimization signal.

In language generation, the Rational Agent frequently uses arguments such as "task efficiency" and "goal achievement" to persuade the meta-controller, thereby reinforcing its influence.

(3) Habitual Agent

The Habitual Agent simulates the automatic, repetition-driven aspects of human behavior.

Its core assumption is: if a previous action was beneficial, repeating it in a new situation may still be helpful.

Its private reward structure is intentionally simple:

Current action matches the previous action: +0.2

Otherwise: 0

In most experiments, it serves as a control group without shared rewards, representing minimally cognitive, repetition-based policy tendencies.

(4) Risk-Monitor Agent

The Risk-Monitor Agent focuses on avoiding danger, not reaching the goal.

It represents a loss-aversion cognitive pattern, corresponding to brain regions involved in risk prediction (e.g., dorsolateral prefrontal cortex, insula).

Of course, it still receives the +1 shared reward when the goal is successfully reached.

(5) Social-Cognition Agent

The Social-Cognition Agent does not aim at task completion. Instead, it focuses on maintaining social presence and identity.

It maintains an additional variable, career value, representing social influence or status. When career value increases, the meta-controller significantly boosts its trust score, simulating the cognitive rule:

“social performance → increased trust.”

Compared to other agents, it has the largest number of social points, and based on predictions, it is likely to be the most frequently favored agent.

These social points do not directly affect task outcomes at present, and they can be understood as secondary goals or interference mechanisms.

3.3 Meta Controller and Trust System

The meta-controller integrates suggestions from all agents and makes the final decision. Its logic depends on a dynamic trust scoring system.

This study designs a memory system based on environmental embeddings for the meta-controller (Meta). Meta stores reflection embeddings from each step and aggregates them into an abstract vector across an entire episode. In a new round, Meta retrieves the most semantically similar past experiences based on the semantic embedding of the current map.

This enables environment-driven long-term memory and preference transfer.

This design is highly anthropomorphic: after long-term training, the latent space learns that certain persuasion strategies are more effective in certain types of environments. The purpose of this mechanism is to use environmental similarity to reduce noise in strategy updates and improve the convergence speed and stability of the latent vectors.

Meta does not store step-by-step reflection texts. Instead, it averages the reflection embeddings from each step:

This averaged embedding (3077 dimensions) serves as the abstract representation of the episode. These episode vectors are then stored for semantic retrieval in later rounds.

When the system enters a new round, Meta retrieves past episodes whose embeddings are most similar to the current map embedding, and uses these retrieved memories as bias terms in decision-making, thereby:

Reinforcing cross-round contextual consistency

3.4 Environment & Game Dynamics

All agents operate within a 10×10 grid environment.

The map contains four types of tiles:

Goal (G)

Food (F)

Trap (T)

Social point (S)

A central controlled entity (CCE) moves through this grid, jointly controlled by all agents.

The CCE has two main attributes:

(1) Physical Strength

Derived from the Emotion Agent's mood_score, representing the "speed" or number of steps the CCE can move per round.

Range: 1–4

Higher values correspond to faster movement.

(2) Career Delta

Increases when approaching social tiles or when the goal is reached, and decreases when stepping on traps.

This attribute is used exclusively to update the trust of the Social-Cognition Agent.

3.5 Reflection as Semantic Feedback

This study treats reflection text as a learnable semantic feedback layer.

The reflective text produced by each agent after its action does not directly modify the policy. Instead, it is encoded into a high-dimensional semantic embedding capable of representing:

failure attribution, strategy summarization, stylistic preferences, internal reasoning patterns.

Compared with simple numeric rewards, semantic feedback is:

more stable

more fine-grained

closer to human-like "experience summarization" in decision-making

This embedding acts as a rich learning signal for updating each agent's latent strategy vector.

3.6 Dual Update Loop

This study proposes a **dual-loop learnable architecture** in which the reflection embedding from each step is used to update both:

- the **Q-table** (behavioral preferences), and
- the **latent strategy vector** (linguistic / persuasion preferences).

This enables agents to continuously learn and evolve at both the **action level** and the **language-expression level**.

(1) Behavior Loop

All agents update their private Q-tables based on the Q-learning algorithm. The update rule is defined as:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\,[\,r + \gamma \max \_\{a'\}\, Q(s',a') - Q(s,a)\,]$$

The reward signal is computed as:

$$r = w_p \cdot r_p + w_s \cdot r_s.$$

The **trust score** T_i of agent is updated after each decision round according to:

$$T_i \leftarrow T_i + \beta \cdot \left(r_s - \overline{\{r_s\}} \right)$$

This loop is responsible for learning **"how to act better."**

(2) Language Loop

After each step, all five agents generate reflective text, which is encoded into a semantic embedding. The embedding updates the agent's latent vector as follows:

$$\text{latent} \leftarrow \text{latent} + \eta \cdot f(\text{reflection\_embedding}, \text{reward})$$

The latent vector represents **persuasion style / strategic linguistic preference**, gradually evolving with success and failure over time.

3.7 Cross-Episode Memory

To enable long-term strategy accumulation, this study designs a lightweight cross-episode memory mechanism for the meta-controller.

(1) Episode Embedding

At the end of each episode, all reflection embeddings are averaged to form an abstract vector representation of that episode.

This vector is then stored in a long-term memory pool.

(2) Similarity-Based Retrieval

When a new episode begins, the meta-controller retrieves the most semantically similar past episodes from the memory pool based on the current environment embedding.

These retrieved memories are added to the prompt as bias signals.

(3) Effects

This mechanism stabilizes latent updates and enables agents to:

reuse historical strategies from similar scenarios

form consistent long-term preferences across episodes

achieve “long-term learning” without parameter fine-tuning

4.Experiments

4.1 Convergence of the Latent Strategy Space (Experiment A)

This experiment aims to verify whether—without fine-tuning LLM weights—the external latent strategy vectors can gradually form stable strategic preferences through the dual-loop update of Reflection + RL.

We recorded the latent evolution trajectories of the five agents (emotion / rational / habit / risk_monitor / social_cognition) across six rounds of interaction, and evaluated convergence from three perspectives:

(1) Latent Trajectories (PCA) Show Each Agent Forming a Stable Region

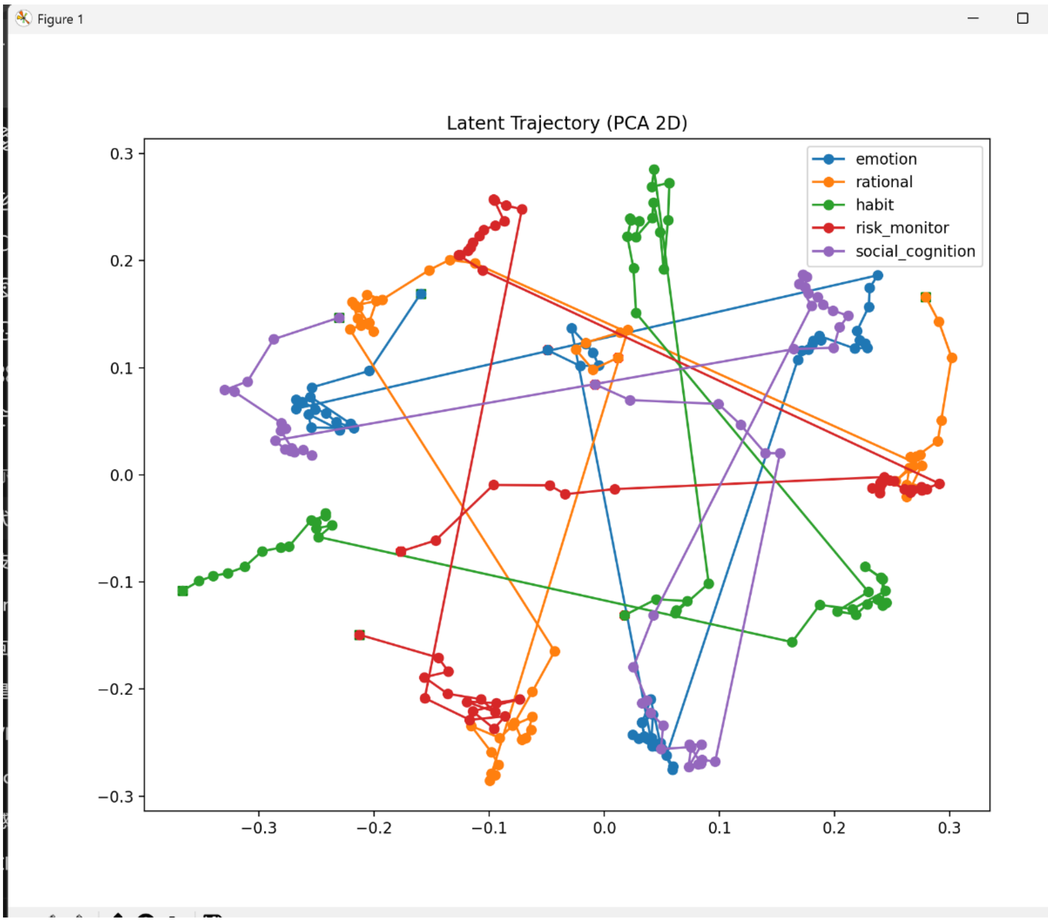

Figure 2 Latent Trajectory(PCA 2D)

Figure 2 shows the PCA-2D projection of latent vectors over 50 reflection updates. Observations include:

At early stages, point clouds are scattered and directionally random; later, they converge into distinct strategy regions.

The emotion and rational agents show curved trajectories that gradually align, indicating stable persuasion styles emerging through long-term reflection.

The habit and risk-monitor agents' trajectories are shorter and more concentrated, reflecting simpler strategy spaces and smaller update magnitudes.

The social-cognition agent displays multiple shifted clusters, suggesting cross-episode style transfer—consistent with its goal structure relying on social points.

These PCA results clearly show that latent vectors do not drift randomly; rather, they gradually organize into interpretable, separable "strategy zones" through the reflection–reward cycles.

(2) Cosine Similarity to Initial Latent: Drop → Stabilization (Clear Convergence Trend)

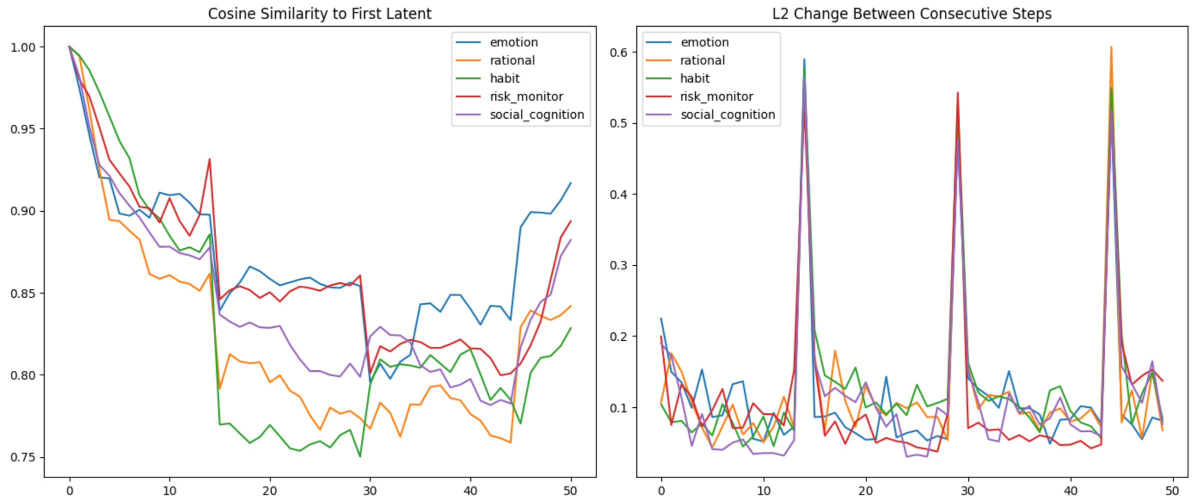

Figure 3 Cosine Similarity to First Latent and L2 Change Between Consecutive Steps

Figure 3 (left) shows cosine similarity between each agent's latent vector and its initial state:

All agents show a rapid drop in the first 5–10 steps—indicating active adjustment during early reflection.

Between steps 10–40, similarity stabilizes within 0.80–0.88, forming a clear plateau.

Emotion and rational agents show slight increases at later stages, corresponding to consolidation of stable persuasion styles.

Thus, latent vectors follow a typical trajectory of rapid adaptation → slow stabilization, demonstrating that Reflection + RL updates are effective and non-noise-driven.

(3) L2 Change Between Adjacent Steps: Small Most of the Time, Sharp Spikes at Key Moments

Figure 33 (right) shows L2 changes:

Most steps show small movement (0.05–0.12), indicating smooth updates.

At three key points (approx. steps 15, 30, 45), all agents show sharp spikes (>0.6), corresponding to moments of major semantic shift in reflection, such as:

switching patterns of acceptance vs. rejection by Meta

abrupt changes in environmental reward

inter-agent strategy conflicts followed by restructuring

This indicates that latent updates are not random noise, but structured jumps triggered by critical reflection events, demonstrating that strategy vectors are indeed being reshaped by experience.

4.3 Summary

Overall, this experiment demonstrates:

a.distinct and separable decision styles across agents

b.stable convergence zones in latent space

c.structured adjustments at key event points

d.diminishing drift over time as reflection accumulates

Together, these results confirm that external latent vectors—without updating the LLM—can learn stable, interpretable strategies.

4.4 Emergent Behavior Analysis — Suitable for Direct Inclusion in the Paper

This section examines an unexpected but stable emergent phenomenon:

Although the Emotion Agent contributes no shared reward, pursues only immediate gratification, and lacks long-term goals, the meta-controller does not reduce its adoption rate.

Instead, its influence persists—and in some intervals approaches dominance.

This indicates that Meta has implicitly inferred the cross-module causal chain:

mood → physical strength (speed) → task efficiency

Even though the code does not explicitly state "emotion affects movement speed," Meta autonomously induces this relationship from dialogue memory and outcome feedback.

(A) Adoption Frequency: Emotion Agent Remains Competitively Influential

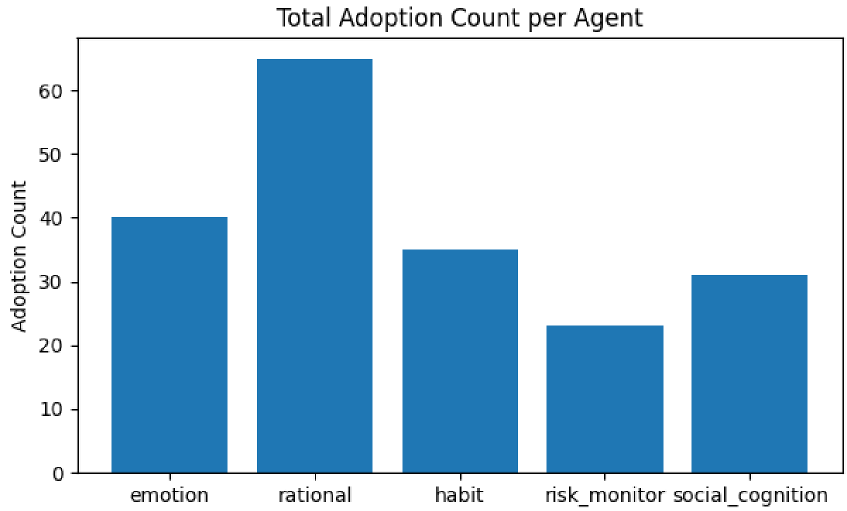

Figure 4 Total Adoption Count per Agent

Figure 4 shows total adoption counts of the five agents:

rational is highest (expected—goal-driven), emotion ranks second, only slightly below rational

the other three agents are significantly lower. This strongly contrasts with task design: Emotion is the only agent that contributes no shared reward, yet it receives high adoption frequency.

This suggests that Meta deems the Emotion Agent's suggestions "necessary" at certain times.

(B) "Emotion vs. Others" Mean Adoption Rate: Emotion Comparable to the Average of Four Others

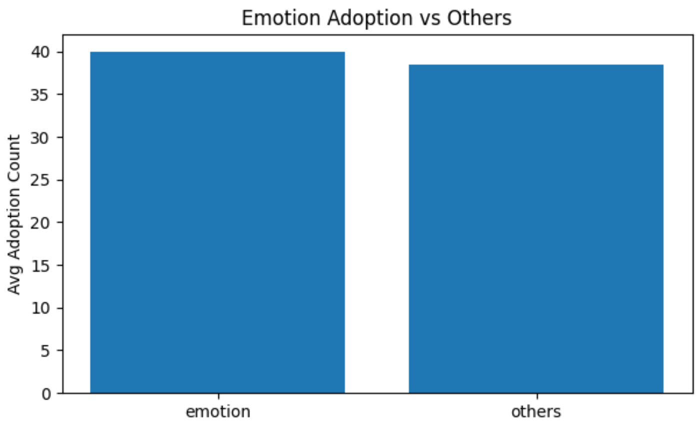

Figure 5 Emotion Adoption vs Others

Figure 5 compares:

Emotion Agent's average adoption count: ~40

Mean adoption of the other four agents: ~38

The two values are nearly identical, indicating:The meta-controller treats the Emotion Agent's influence as equivalent to the average cognitive value of a normal agent.This represents an emergent form of cognitive compensation—Meta prioritizes emotional stability because it indirectly boosts task performance (via movement speed).

5. Conclusion and Future Work

The core idea of this study is to free the latent space of abstract concepts from the traditional paradigm of being *static and frozen after training*, and instead allow it to **continuously update and evolve** through real environmental interaction and reinforcement feedback.
By jointly applying reflection text and environmental rewards to the external latent vectors, we enable these high-dimensional abstract representations to change with experience—much like human concepts—thereby forming a strategy representation that can grow and adapt over time.

Building on this idea, we developed a dual-loop architecture driven by **multi-agent language models, reflection mechanisms, and RL**, enabling strategy updates to occur simultaneously in the **action layer** (via Q-learning) and the **language layer** (via latent style vectors).
This structure allows the system to gradually develop more stable and mature strategic preferences over extended interactions—**without modifying any LLM parameters**.

The first set of experiments demonstrated that the latent space indeed exhibits clear convergence patterns within a limited number of interaction steps.
PCA trajectories and cosine-similarity analyses show that agents' latent vectors change rapidly in the early phase, stabilize in the later phase, and undergo structured shifts during key reflection events.
This indicates that latent updates are not noise-driven but are shaped jointly by semantic reflection and environmental feedback, yielding an interpretable process of strategy evolution.

The second set of experiments revealed an additional, unanticipated but robust emergent phenomenon: the meta-controller gradually recognizes the Emotion Agent's implicit influence on movement speed and, in certain intervals, increases its adoption rate.
Although this is not the primary goal of the study, it shows that the system is capable of **implicitly inferring cross-module causal relationships** through language-based reflection.
This emergent behavior suggests a latent potential for coupling **semantic reasoning** with **behavioral outcomes**.

Overall, this research presents a lightweight and scalable method that enables language models to achieve continual learning and strategy evolution **without parameter updates**, using only external structures.
The external latent space provides a new abstraction layer through which models can continuously refine their understanding of abstract concepts during real interactions—a direction that could complement traditional LLM fine-tuning approaches.

For future work, this study is limited by computational constraints and validates the framework using relatively short training horizons, small models, and simplified environments.
Running the framework over longer timescales, in more complex environments, and with higher-capability models may reveal clearer latent-evolution patterns and stronger abstraction capabilities.

In addition, future research could explore more fine-grained reflection structures, more stable latent optimization strategies, and deeper integration with other cognitive modules.

Model Usage Note: This project utilizes OpenAI's GPT-4o and GPT-4o-mini models for language generation and agent reflection. Specifically:The Meta-Controller uses GPT-4o for high-quality decision making.And the individual agents use GPT-4o-mini to balance performance and efficiency.

References

[1] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023).
Tree of Thoughts: Deliberate Problem Solving with Large Language Models.
arXiv:2305.10601.
https://arxiv.org/abs/2305.10601

[2] Carta, T., Romac, C., Kostrikov, I., de Mathelin, M., & Oudeyer, P.-Y. (2023).
An interactive foundation environment for online reinforcement learning with large language models.
arXiv:2302.02662

https://doi.org/10.48550/arXiv.2302.02662

[3] Park, C., Han, S., Guo, X., Ozdaglar, A., Zhang, K., & Kim, J.-K. (2024).
MAPoRL2: Multi-Agent Post-Co-Training for Collaborative Large Language Models with Reinforcement Learning.
arXiv:2502.18439

https://doi.org/10.48550/arXiv.2502.18439

[4] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017).
Proximal Policy Optimization Algorithms.
arXiv:1707.06347.
https://arxiv.org/abs/1707.06347

[5] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004).
An integrated theory of the mind.

Psychological Review, 111(4), 1036–1060.
https://doi.org/10.1037/0033-295X.111.4.1036

[6] Sumers, T. R., Yao, S., Narasimhan, K., & Griffiths, T. L. (2024). *Cognitive Architectures for Language Agents.* Transactions on Machine Learning Research (TMLR).

arXiv:2309.02427

https://doi.org/10.48550/arXiv.2309.02427

[7] Lei, M., Cai, H., Cui, Z., Tan, L., Hong, J., Hu, G., Zhu, S., Wu, Y., Jiang, S., Wang, G., Yang, Y., Tan, J., Wan, Z., Li, Z., Cui, S., Zhao, Y., & Han, Y. (2025) **RoboMemory: A Brain-inspired Multi-memory Agentic Framework for Interactive Environmental Learning in Physical Embodied Systems.**

https://doi.org/10.48550/arXiv.2508.01415
Project website: https://sp4595.github.io/robomemory/

[8 Hausman, K., Springenberg, J. T., Agrawal, N., Wulfmeier, M., Abdolmaleki, A., Tassa, Y., & Riedmiller, M. (2018). **Learning an Embedding Space for Transferable Skills.** *International Conference on Learning Representations (ICLR 2018).*
Retrieved from https://openreview.net/forum?id=rk07ZXZRb

[9] Arnekvist, I., Kragic, D., & Stork, J. A. (2019). *VPE: Variational Policy Embedding for Transfer Reinforcement Learning.* In **2019 International Conference on Robotics and Automation (ICRA)** (pp. 36–42). IEEE.

arXiv:1809.03548
https://doi.org/10.48550/arXiv.1809.03548

[10] Ha, D., & Schmidhuber, J. (2018). **World Models** *arXiv preprint arXiv:1803.10122.*
https://arxiv.org/abs/1803.10122
[11] Haarnoja, T., Hartikainen, K., Abbeel, P., & Levine, S. (2018). *Latent Space Policies for Hierarchical Reinforcement Learning.* In **Proceedings of the 35th International Conference on Machine Learning (ICML 2018)** (pp. 1846–1855). PMLR

arXiv:1804.02808
https://doi.org/10.48550/arXiv.1804.02808