Research Article

Mutual Influence AI: Trust-Based Cooperation Mechanisms for LLM Multi-Agent Systems

Vaclav OUJEZSKY[1,*], *Pavel NOVAK*[1]

[1]Department of Computer Systems and Communications, Faculty of Informatics, Masaryk University, Botanicka 68a, Brno, Czechia

oujezsky@fi.muni.cz, novakpav@mail.muni.cz

[*]Corresponding author: Vaclav Oujezsky; oujezsky@fi.muni.cz

DOI: 10.15598/aeee.v23i4.250910

Article history: Received Sep 12, 2025; Revised Oct 22, 2025; Accepted Nov 16, 2025; Published Dec 31, 2025.

Abstract. *This paper introduces Mutual Influence AI, a novel concept for adaptive cooperation in multi-agent systems. Unlike classical independent reasoning or centralized orchestration, our approach introduces an explicit mutual influence factor μ that captures trust-adjusted peer feedback and directly modulates large language model (LLM) generation. We present (i) a mathematical formalization of mutual influence, (ii) a prototype implementation integrated with Microsoft AutoGen for LLM-based agents, and (iii) qualitative evidence that the framework improves adaptability, transparency, and coordination in multi-agent dialogues. Results show that Mutual Influence AI stabilizes group interactions efficiently while providing interpretable control over how agents influence each other. This positions Mutual Influence AI as a new paradigm for LLM-driven multi-agent systems with potential applications ranging from collaborative problem solving to cybersecurity. Quantitatively, across 167 simulation runs, cross–role agreement increased from* 0.19 *(baseline) to* 0.50 *under influence (approx. +160%), with median revision depth (approx. 1.0). Under adversarial feedback, agreement still improved (0.18 to 0.47).*

Keywords

Mutual influence, multi-agent systems, large language models, AutoGen.

1. Introduction

Multi-agent systems (MAS) have become a cornerstone of modern artificial intelligence, with applications in autonomous robotics, cyber defense, decision support, and large-scale knowledge management. A central challenge in these systems is achieving robust coordination among agents in dynamic environments, where objectives may evolve and peer performance may vary over time. Commonly used coordination mechanisms in multi-agent frameworks assume fully independent reasoning, which may leads to weak cooperation, or they rely on centralized control, which introduces scalability limits and single points of failure [1].

Large language models (LLMs) enable multi-agent problem solving where several role-specialized agents exchange feedback and iteratively refine a shared solution [2]. Existing orchestration frameworks such as *AutoGen* [3] make this practical, yet a key limitation remains [4]: agents typically do not adapt the extent to which they rely on peers according to peer reliability. As a result, systems either over-align (premature consensus, "groupthink") or remain overly independent (missed cooperation gains). We address this gap by introducing an explicit, interpretable notion of *mutual influence*. Each agent maintains peer-specific trust scores and aggregates them into a scalar influence factor $\mu \in [0, 1]$ that *directly* modulates the agent's generation: (i) via a logistic mixture hint $\lambda(\mu)$ inserted into the prompt, and (ii) via temperature scaling $T(\mu)$ in the LLM client. The mechanism is lightweight, requires no model retraining, and integrates cleanly with AutoGen [3].

Problem setting: We consider a team of n LLM agents interacting in discrete rounds $t = 0, 1, \ldots$. Agent i maintains trust $s_{i \leftarrow j}(t) \in [0, 1]$ for each peer $j \neq i$ and updates it by an exponential moving average (EMA) with rate $\beta \in (0, 1]$. The mutual influence of agent i is $\mu_i(t) = \frac{1}{n-1} \sum_{j \neq i} s_{i \leftarrow j}(t)$. We map μ_i to two modulation functions: (a) a logistic mixture coefficient $\lambda_i(t) = \sigma(k(\mu_i(t) - \tau))$ with steepness $k > 0$ and threshold $\tau \in (0, 1)$, and (b) a temperature schedule $T_i(t) = \frac{T_0}{1 + \alpha\, \mu_i(t)}$ with $\alpha \geq 0$. The tuple $[\mu_i, \lambda_i, T_i]$ is exposed in the prompt prefix and the temperature of the client, yielding transparent, controllable influence.

We further provide an ablation over (β, k, τ, α) and analyze normal vs. adversarial settings to elucidate both benefits and failure modes of influence-driven coordination.

Contributions:

- *Explicit, interpretable influence.* We formalize a trust-adjusted *mutual influence* μ and show how to inject it into LLM generation through $\lambda(\mu)$ and $T(\mu)$, enabling a tunable balance between peer alignment and independent reasoning.
- *Lightweight integration.* We implement a prototype on AutoGen v0.4 that needs no model retraining and works as a drop-in orchestration layer.
- *Evaluation protocol.* We propose simple, automatable metrics for multi-agent coordination: *AgreementRate* (Jaccard similarity of agents' key points), *Rounds-to-Approval* (convergence speed), and *RevisionDepth* (magnitude of change).
- *Quantitative evidence.* In Section 4. we provide a quantitative comparison against a no-influence baseline and ablations over (β, k, τ, α), plus convergence traces of μ.

Notation of hyperparameters: We consistently use β for the EMA update rate, k for the logistic steepness, τ for the logistic threshold, and α for the temperature-scaling strength. These are introduced here and referenced in later sections to avoid ambiguity.

2. State of the Art

Research on inter-agent influence spans multiple lines of work in multi-agent learning and, more recently, LLM-based orchestration. Prior methods promote coordination either implicitly (via intrinsic rewards or action-prediction coupling) or through centralized modules, but typically lack an explicit, prompt-visible influence signal that can be tuned at run time without retraining. In contrast, our approach exposes a scalar μ and its mapped controls $\lambda(\mu)$ and $T(\mu)$ directly in the LLM loop, providing transparency and immediate adjustability.

The role of influence between agents has been studied in multiple strands of multi-agent reinforcement learning (MARL). One prominent line of work is *Mutual-Help MARL* proposed by Qiu et al. [5], where agents anticipate peer actions and incorporate imitation mechanisms to promote cooperative behavior. This approach highlights the value of explicitly modeling inter-agent dependencies, although it remains mostly tied to action prediction rather than flexible adaptation. Our work extends this principle beyond action coupling to language-based reasoning, where inter-agent dependency is expressed through a continuous trust variable μ that affects the generation process rather than a policy update.

Another perspective is offered by Wen [6], who models mutual influence through recursive reasoning and policy-space meta-game analysis. Here, the emphasis lies on hierarchical reasoning about others' policies, enabling agents to anticipate deeper levels of interaction. While powerful, this method is computationally heavy and less suitable for lightweight, real-time coordination. We adopt the conceptual insight of mutual influence as recursive dependence but implement it as a closed-form modulation, allowing LLM agents to achieve similar adaptability at negligible cost.

A different direction was taken by Jaques et al. [7], who introduced *social influence as intrinsic motivation*. In their formulation, agents are rewarded for the causal impact of their actions on the behavior of others. This brings a novel incentive structure to cooperative MARL, but it still requires carefully designed reward shaping and does not explicitly expose interpretable influence parameters. Our method replaces the hidden intrinsic-reward signal with an explicit, interpretable influence coefficient μ that each agent can observe and control during generation.

More recently, Du et al. [8] proposed *situation-dependent causal influence* (SCIC), which adapts intrinsic rewards according to contextual factors. This approach allows for context-sensitive cooperation and improves exploration efficiency, yet again it operates primarily through intrinsic reward design rather than explicit trust-based modulation. Mutual Influence AI generalizes this notion by introducing a unified modulation layer that links context, trust, and sampling behavior through $\lambda(\mu)$ and $T(\mu)$, making the adaptation mechanism both observable and directly tunable.

Recent work on the Trust-Vulnerability Paradox [9] highlights the risks of excessive inter-agent trust. In

contrast, our framework focuses on controlled, interpretable modulation of peer influence, aiming to balance cooperation and independence.

Recent work on *LLM Influence Networks* [10] models directed information flow between agents as a weighted graph W_{ij}, enabling global control over inter-agent dependencies. Our formulation is complementary: rather than estimating a full influence matrix, *Mutual Influence AI* maintains a local scalar reputation μ_i per agent, derived from peer-specific trust scores and updated via exponential moving averages.

Liang et al. [11] propose a Bayesian trust-estimation framework for coordination among LLM agents in industrial settings. Their approach updates posterior trust based on observed reliability, serving a complementary goal to our EMA-based influence formulation, which focuses on lightweight, real-time adaptation and decentralized orchestration.

Prior research demonstrates that modeling influence among agents can enhance cooperation, exploration, and coordination. However, most existing methods rely on either implicit incentives or complex reasoning frameworks, which often obscure how influence is represented and adjusted. This motivates our proposal of *Mutual Influence AI*, where the influence factor μ is made explicit, mathematically defined, and directly integrated into the agent's decision-making loop. Unlike previous work, our approach emphasizes interpretability, transparency, and real-time adaptability, while also demonstrating practical implementation within LLM-based multi-agent orchestration.

2.1. Positioning w.r.t. LLM orchestration frameworks

Agent frameworks such as Communicative Agents for "Mind" Exploration of Large Scale Language Model Society (CAMEL) [12] [13], SWE-Agent [14], and Toolformer [15] demonstrate the utility of structured multi-agent or tool-augmented prompting. Our method is complementary: rather than introducing new roles or tools, we add an *explicit influence control* that can be layered on top of these frameworks. Concretely, our AutoGen-based prototype shows how μ can modulate both prompt content (via $\lambda(\mu)$ hints) and sampling behavior (via $T(\mu)$), which—unlike prior orchestration logic—exposes trust dynamics as first-class, interpretable signals. In this way, our work extends orchestration frameworks by providing an explicit, mathematically grounded trust mechanism that can be observed, logged, and adjusted at run time—something absent from prior LLM coordination systems.

2.2. Novelty of Mutual Influence AI

The proposed framework of *Mutual Influence AI* extends existing approaches to multi-agent cooperation by making the notion of influence both explicit and operational. At the core of the method lies a decentralized, prompt-based mechanism for LLM agents, where the influence factor μ directly modulates the generation process. Unlike implicit incentive structures or black-box coordination schemes, our approach introduces interpretable parameters — specifically the mixture coefficient $\lambda(\mu)$ and the temperature function $T(\mu)$ — that allow the balance between peer alignment and independent reasoning to be adjusted in a transparent manner.

Another distinctive aspect of our work is the integration with AutoGen, which provides a lightweight and flexible framework for orchestrating LLM agents. Through this integration, peer feedback can be incorporated in real time without relying on a centralized controller. This makes the system scalable, robust to single-point failures, and adaptable to dynamically changing conditions, which is particularly relevant in domains such as cybersecurity where new information and adversarial behavior must be handled immediately.

Finally, Mutual Influence AI introduces an unusual level of transparency into the dynamics of multi-agent interaction. Because the influence factor μ is both mathematically defined and explicitly exposed to the agents' prompts, it becomes possible to monitor and steer the process of consensus formation. Agents can dynamically adjust the degree to which they conform to or diverge from their peers, thus avoiding the extremes of blind consensus on the one hand and complete disregard of peer input on the other. This interpretability not only facilitates system-level analysis and debugging but also offers a foundation for principled control of trust and influence in complex multi-agent environments.

In the next section, we formalize how the scalar influence factor μ governs both peer-mixing and temperature modulation in practice.

3. Mutual Influence: Mathematical Formulation

Let $\mathcal{A} = \{1, \ldots, n\}$ denote the set of agents and $t = 0, 1, 2, \ldots$ the discrete dialogue rounds.

Each agent i maintains a peer-specific trust score:

$$s_{i \leftarrow j}(t) \in [0, 1], \quad j \neq i, \tag{1}$$

quantifying how much agent i values the contribution of peer j after round t. The *mutual influence* of agent

i at time t is defined as the mean trust that i places in its peers:

$$\mu_i(t) = \frac{1}{|\mathcal{A}| - 1} \sum_{j \in \mathcal{A} \setminus \{i\}} s_{i \leftarrow j}(t). \tag{2}$$

After observing responses in round t, each agent i provides a feedback signal $\phi_{i \leftarrow j}(t) \in [0, 1]$ reflecting the perceived usefulness or correctness of peer j's output. Trust is then updated via exponential moving average (EMA):

$$s_{i \leftarrow j}(t+1) = (1 - \beta)\, s_{i \leftarrow j}(t) + \beta\, \phi_{i \leftarrow j}(t), \tag{3}$$

where $\beta \in (0, 1]$ is the update rate*.

Two modulation functions are derived from μ_i:

- A logistic mixture coefficient

$$\lambda_i(t) = \sigma\big(k(\mu_i(t) - \tau)\big) = \frac{1}{1 + e^{-k(\mu_i(t) - \tau)}}, \tag{4}$$

  where $k > 0$, $\tau \in (0, 1)$, used as a *textual hint* in the LLM prompt.

- A temperature scaling

$$T_i(t) = \frac{T_0}{1 + \alpha\, \mu_i(t)}, \quad \alpha \geq 0, \tag{5}$$

  which directly adjusts the sampling temperature of the LLM client.

The final prompt at round t is constructed as:

$$\text{prompt}_i(t) = [\mu_i(t),\, \lambda_i(t),\, T_i(t)] \parallel \text{user task}. \tag{6}$$

3.1. Derived Evaluation Metrics

Following prior work on cooperative multi-agent evaluation and dialogue consistency, we adopt a small set of interpretable metrics that capture inter- and intra-role alignment, revision dynamics, and convergence properties. Let $P_F(t)$ and $R_F(t)$ denote the sets of streaming features proposed by Planner and Researcher at round $t \in \{1, 2\}$. We define:

Cross–role agreement: To quantify the semantic overlap between the two roles, we use the Jaccard index, commonly applied in multi-agent cooperation studies, following the consensus evaluation used in Multi-agent Debate frameworks [16]:

$$A_{\text{cross}}(t) \;=\; J\big(P_F(t), R_F(t)\big) \;=\; \frac{|P_F(t) \cap R_F(t)|}{|P_F(t) \cup R_F(t)|}. \tag{7}$$

*(The update rule (Eq. 3) is directly implemented in the orchestration loop (Listing 4).

Intra–role stability: Following consistency metrics for LLM-based agents, we assess how stable each role remains across rounds:

$$\begin{aligned} S_{\text{planner}} &= J\big(P_F(1), P_F(2)\big), \\ S_{\text{researcher}} &= J\big(R_F(1), R_F(2)\big). \end{aligned} \tag{8}$$

This measures whether an agent maintains internal coherence as it adapts to peer feedback.

Revision depth: To estimate how much content changed between iterations, we define a revision–based metric inspired by dialogue revision graphs:

$$D_{\text{rev}} \;=\; \big|\, \big(P_F(2) \cup R_F(2)\big) \setminus \big(P_F(1) \cup R_F(1)\big) \big|. \tag{9}$$

Larger D_{rev} values correspond to deeper structural edits or significant content additions.

Optionally, we also suggest a *canonical overlap* on a fixed tag alphabet $\mathcal{C}$ (e.g., `{flow_bytes, packets, protocol, ...}`):

$$C_{\text{can}}(t) \;=\; \frac{|\,\text{tags}(P(t)) \cap \text{tags}(R(t))\,|}{|\,\text{tags}(P(t)) \cup \text{tags}(R(t))\,|}. \tag{10}$$

These metrics can be automatically computed from structured AutoGen logs and provide interpretable evidence of inter-agent coordination, revision efficiency, and prompt-level adaptation.

3.2. Mapping to the Current Prototype (AutoGen-optional)

Our experiments use a lightweight orchestrator that mimics the AutoGen-style loop but runs without a group-chat controller. Each role is implemented as an instance of `MutualInfluenceAssistant` that (i) exposes a scalar influence μ, (ii) accepts explicit peer feedback via EMA updates, and (iii) modulates both the prompt prefix and temperature according to (k, τ, α).

Instantiation: Each agent is constructed with a role tag, a short system hint, and modulation hyperparameters (k, τ, α), as shown in Listing 1. This defines three core roles: a planner, a researcher, and a critic, corresponding to the functions described in Section 4. .

Influence state: Each assistant maintains peer-specific trust scores $s_{i \leftarrow j}$ and exposes the overall influence μ_i as their mean reputation across peers. Trust is updated using an exponential moving average (EMA) according to Eq. (3), ensuring that influence remains responsive to feedback while preserving stability across dialogue rounds. The corresponding implementation is

Listing 1: Assistant construction (current prototype).

```
planner    = MutualInfluenceAssistant(
    "planner",    "Role: Planner. Short
        ↪ prioritized plan.",
    API_KEY, k, tau, alpha
)
researcher = MutualInfluenceAssistant(
    "researcher", "Role: Researcher. List
        ↪ concise streaming features and
        ↪ brief reasons.",
    API_KEY, k, tau, alpha
)
critic     = MutualInfluenceAssistant(
    "critic",     "Role: Critic. Point
        ↪ gaps/risks. Reply 'APPROVE' if
        ↪ sufficient.",
    API_KEY, k, tau, alpha
)
```

Listing 2: Computation of influence factor μ (Eq. 2) and trust update via EMA (Eq. 3) (unchanged conceptually).

```
# μ is computed on the fly from peer_scores
@property
def mu(self) -> float:
    if not self.peer_scores:
        return 0.5
    return sum(self.peer_scores.values())
        ↪ / len(self.peer_scores)

def receive_feedback(self, from_peer: str,
    ↪ score: float, beta: float = 0.6):
    old = self.peer_scores.get(from_peer,
        ↪ 0.0)
    self.peer_scores[from_peer] = (1.0 -
        ↪ beta) * old + beta * float(score)
```

shown in Listing 2, and the feedback values $\phi_{i\leftarrow j}(t)$ are supplied through the `receive_feedback(source, value, beta)` calls within the orchestration loop (Listing 4), where `value` represents the normalized agreement or approval score provided by peer j.

Prompt and temperature modulation: Calls differentiate between baseline (uninfluenced) and influence-enabled rounds. The assistant composes a transparent prefix containing $(\mu, \lambda(\mu), T(\mu))$ and scales the temperature according to $T(\mu) = T_0/(1+\alpha\mu)$ and $\lambda(\mu) = \sigma(k(\mu-\tau))$, as illustrated in Listing 3.

Two-round orchestrator: The main experimental loop, shown in Listing 4, executes one baseline round and one influence-enabled round. After each baseline interaction, peer feedback is applied through EMA updates, and agents generate influenced responses using the updated μ values.

AutoGen compatibility: Although our runs use the minimal loop above, the same `MutualInfluenceAssistant` can be directly plugged into AutoGen v0.4. In that case, the group-chat driver (e.g., `RoundRobinGroupChat`) replaces the hand-written loop, while μ-based prefixing and temperature scaling remain unchanged. This makes the influence layer *framework-agnostic* and easily extensible to multi-turn AutoGen conversations.

4. Experimental Plan and Metrics

We evaluate the proposed influence mechanism in two regimes within a single security-flavored reasoning task involving three roles (planner, researcher, critic): a normal regime and an adversarial regime simulating noisy feedback. We compare ours (μ-on) against a *No-Influence* baseline, and report ablations over (β, k, τ, α).

Protocol: Each run consists of two rounds: *Round 1 (baseline)* without influence modulation and *Round 2 (influence)* with μ–driven prompting and temperature scaling. Agents output exactly one JSON object per prompt; features are extracted from the `"features"` array. The critic outputs `"decision"` $\in \{$`APPROVE`, `REVISE`$\}$. We evaluate both a *normal* regime and an *adversarial* regime (the latter simulates

Listing 3: Single-call interface with modulation of $\lambda(\mu)$ and $T(\mu)$ according to Eqs. (4)–(5).

```
async def call(self, user_task: str,
    ↪ influenced: bool, base_temp: float =
    ↪ 0.7) -> str:
    if not influenced:
        # Round 1: baseline (no modulation)
        return await
            ↪ self.run(task=user_task,
            ↪ temperature=base_temp)

    # Round 2: influence on
    mu  = self.mu
    lam = 1.0 / (1.0 + math.exp(-k * (mu -
        ↪ tau)))      # sigma(k*(mu - tau))
    temp = base_temp / (1.0 + alpha * mu)
        ↪              # T0/(1 + alpha*mu)

    prefix = f"[mutual_influence
        ↪ mu={mu:.2f}, lam={lam:.2f},
        ↪ T={temp:.2f}] "
    task   = prefix + "\n\n" + user_task
    return await self.run(task=task,
        ↪ temperature=temp)
```

Listing 4: Two-round orchestration applying peer feedback $\phi_{i \leftarrow j}$ and updating influence μ according to Eq. (3). (no group-chat controller).

```
# Round 1: baseline (uninfluenced)
p1 = await planner.call(
    P["planner_baseline"],
    influenced=False
)
r1 = await researcher.call(
    P["researcher_baseline"],
    influenced=False
)
c1 = await critic.call(
    P["critic_baseline"],
    influenced=False,
    base_temp=0.2
)

# Feedback wiring (normal vs. adversarial
    ↪ regimes)
if adversarial:
    planner.receive_feedback("critic",    0.1,
    ↪ beta)
    researcher.receive_feedback("critic", 0.1,
    ↪ beta)
else:
    planner.receive_feedback("critic",    0.9,
    ↪ beta)
    researcher.receive_feedback("critic", 0.7,
    ↪ beta)

planner.receive_feedback("researcher", 0.8,
    ↪ beta)
researcher.receive_feedback("planner",  0.85,
    ↪ beta)
critic.receive_feedback(
    "planner",
    (0.8 if not adversarial else 0.4),
    beta
)
critic.receive_feedback(
    "researcher",
    (0.75 if not adversarial else 0.4),
    beta
)

# Round 2: influenced (\mu active in prefix +
    ↪ temperature)
p2 = await planner.call(
    P["planner_influenced"],
    influenced=True
)
r2 = await researcher.call(
    P["researcher_influenced"],
    influenced=True
)
c2 = await critic.call(
    P["critic_influenced"],
    influenced=True
)
```

biased/noisy feedback from the critic in the EMA update).

Metrics: We compute: (i) cross–role agreement $A_{\text{cross}}(t)$, (ii) intra–role stability $S_{\text{planner}}, S_{\text{researcher}}$, (iii) revision depth D_{rev}, and (iv) approval presence in each round (APPROVE). Means are reported across seeds and hyperparameters; significance is assessed via paired non–parametric tests.

Tab. 1: Aggregate outcomes across 167 runs (mean rates).

	Baseline	Influence
Cross–role agreement A_{cross} (all)	0.19	0.50
Cross–role agreement A_{cross} (adversarial)	0.18	0.47
Critic approval (normal)	35.8%	39.5%
Critic approval (adversarial)	33.7%	26.7%
Median revision depth D_{rev}	≈ 1.0	

Implementation details: We use AutoGen v0.4. Influence is injected as a textual prefix with $(\mu, \lambda(\mu), T(\mu))$, where $T(\mu) = T_0/(1+\alpha\mu)$ and $\lambda(\mu) = \sigma(k(\mu - \tau))$, with $T_0 = 0.7$. *Hyperparameter sweep:* $\beta \in \{0.3, 0.6, 0.9\}$, $k \in \{3, 6\}$, $\tau \in \{0.4, 0.5, 0.6\}$, $\alpha \in \{0.4, 0.8, 1.2\}$, seeds $\in \{1, 2\}$, both regimes (normal/adversarial). Due to occasional timeouts and post-hoc filtering, not all grid combinations were retained; the final dataset comprises $N = 167$ successful runs (missing combinations appear as white cells in Figure 6), Section 5. .

5. Results and Discussion

5.1. Results

Table 1 summarizes aggregate outcomes across 167 simulation runs under both normal and adversarial regimes. As shown in Figure 1, the overall cross–role agreement A_{cross} markedly increases when influence is enabled, rising from a mean of 0.19 (baseline) to 0.50 (influence). A paired Wilcoxon signed–rank test confirms that this improvement is statistically significant ($W = 276$, $p < 0.001$), with a large effect size

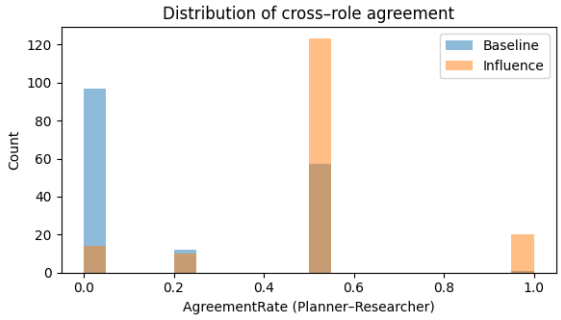

Fig. 1: Distribution of cross–role agreement A_{cross} (Planner–Researcher): baseline vs. influence across all runs.

Tab. 2: Paired statistical tests comparing Baseline vs. Influence across 167 runs. Agreement: Wilcoxon signed–rank with Cliff's δ. Approval: McNemar with continuity correction.

	Baseline	Influence	Test (stat, p)	Effect / Counts
Agreement (all)	0.191	0.500	W=276, p<0.001	$\delta = -0.57$ (large)
Agreement (adversarial)	0.184	0.469	W=138, p<0.001	$\delta = -0.53$ (large)
Approval (normal)	35.8%	39.5%	χ^2=0.08, p=0.78	$\begin{bmatrix} 26 & 26 \\ 23 & 6 \end{bmatrix}$
Approval (adversarial)	33.7%	26.7%	χ^2=0.54, p=0.46	$\begin{bmatrix} 37 & 20 \\ 26 & 3 \end{bmatrix}$

by Cliff's $\delta = -0.57$. To compare the agreement between *Baseline* and *Influence*, we used the paired Wilcoxon signed–rank test and reported Cliff's δ as a non–parametric effect size. Across all runs, the increase in cross–role agreement was significant (Wilcoxon $W = 276$, $p < 0.0001$; Cliff's $\delta = -0.574$, *large*), and remained significant in the adversarial regime (Wilcoxon $W = 138$, $p < 0.0001$; Cliff's $\delta = -0.525$, *large*).[†]

For approval outcomes, see Figure 2, we constructed paired 2×2 tables (approval observed vs. not) and applied McNemar's test with continuity correction [17] separately for normal and adversarial regimes, Table 2. We did not observe a statistically significant change in approval rates (normal: $\chi^2 = 0.082$, $p = 0.775$; adversarial: $\chi^2 = 0.543$, $p = 0.461$). In the normal regime, the approval rate slightly increased from 35.8 % to 39.5 % after enabling influence, suggesting improved planner–researcher alignment. Under adversarial feedback, however, approvals decreased from 33.7 % to 26.7 %, indicating that the influence mechanism correctly dampened trust when peer feedback was unreliable.

Overall, agreement improvements under *Influence* are significant at $p < 0.01$ ($p < 10^{-7}$), with large effect sizes by Cliff's δ, while approval differences are not statistically significant. The pattern confirms that the proposed influence mechanism enhances cross–role consistency without artificially inflating approval frequencies.

[†]The negative sign of Cliff's δ reflects that *Influence* values exceed *Baseline* in our implementation of the statistic (direction: Baseline − Influence).

This demonstrates that the mutual influence mechanism consistently enhances inter–agent alignment across the ensemble of runs. When results are separated by regime (normal vs. adversarial), the same trend is observed in Figure 3, confirming that the mechanism promotes alignment even when part of the feedback is deliberately corrupted. Under normal conditions, influence increased the cross–role agreement from approximately 0.20 to 0.52, whereas in the adversarial setting it still improved from 0.18 to 0.47, indicating that the trust-weighted modulation remains robust to noisy or biased feedback.

To explore how influence strength interacts with modulation hyperparameters, Figure 4 shows the correlation between cross–role agreement A_{cross} and the parameters α (temperature sensitivity) and β (EMA update rate). Each point represents a simulation run. Solid lines show binned means; dashed lines show linear fits. Higher α values, which produce stronger temperature scaling, tend to enhance stability but may reduce diversity, while moderate β values (≈ 0.6) yield the best alignment, balancing responsiveness and smooth adaptation.

The temporal behavior of the mutual influence factor $\mu(t)$ is illustrated in Fig. 5. Across roles (planner, researcher, critic), $\mu(t)$ rapidly converges after one feedback iteration and stabilizes within $\mu \in [0.54, 0.56, 0.57]$ by Round 2. This pattern is consistent with the exponential moving average (EMA) formulation and the bounded prompt-level modulation, confirming that the mechanism introduces fast yet stable adaptation dynamics. All agents start with a neu-

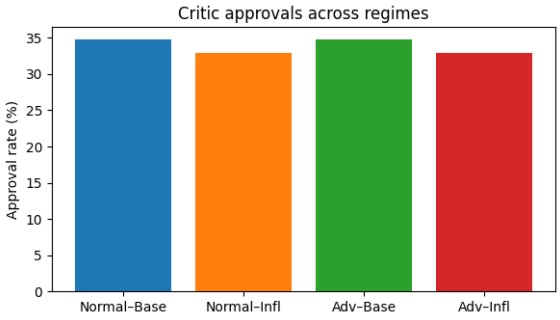

Fig. 2: Approval rates (stacked bars): baseline vs. influence under normal and adversarial regimes.

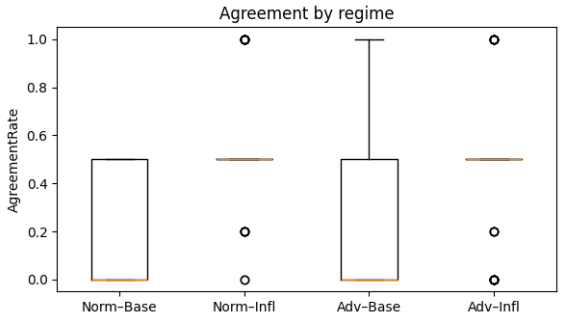

Fig. 3: Agreement by regime: normal vs. adversarial, baseline vs. influence (box/violin illustration).

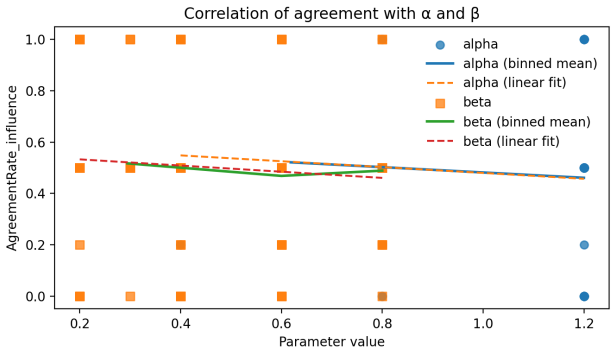

Fig. 4: Correlation of agreement A_{cross} with modulation parameters α and β.

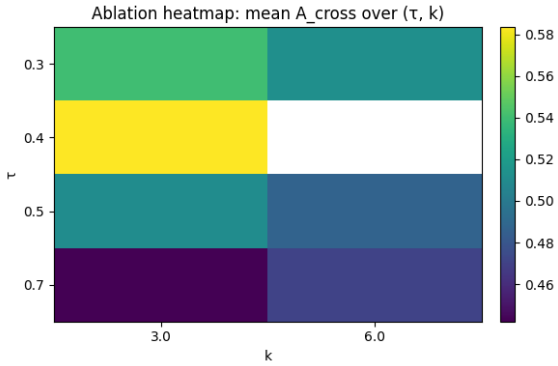

Fig. 6: Ablation heatmap: mean A_{cross} over grid (τ, k) for fixed α, β. The results confirm that Mutual Influence AI remains stable across a wide range of logistic modulation parameters.

tral prior ($\mu \approx 0.50$) in Round 1, corresponding to uninfluenced interaction. After a single feedback iteration, values diverge slightly, with $\mu_{\text{planner}} \approx 0.57$, $\mu_{\text{researcher}} \approx 0.56$, and $\mu_{\text{critic}} \approx 0.54$. This divergence is expected and desirable: each role receives distinct feedback and therefore forms its own trust-adjusted influence level. The Planner, which directly integrates both peer and critic feedback, exhibits the highest update amplitude, while the Critic remains more conservative due to mixed evaluation inputs. Overall, the pattern confirms that the exponential moving average (EMA) update produces rapid yet stable adaptation, leading to near-stationary μ values after only one iteration.

Finally, Figure 6 presents an ablation over (τ, k), visualizing how the logistic threshold τ and steepness k affect mean cross–role agreement. The color intensity encodes the average agreement A_{cross}, with yellow indicating the highest values (≈ 0.58) and purple the lowest (≈ 0.45). White cells denote missing runs in the ablation grid where insufficient data were available. Performance peaks around $\tau \approx 0.4$ and $k \approx 3$, indicating that smoother logistic modulation yields the most coherent alignment between agents. At higher thresholds ($\tau > 0.6$), agents become overly conservative and agreement declines. Overall, results confirm that Mutual Influence AI is robust to a wide range of hyperparameter settings, with stable behavior within $\tau \in [0.45, 0.55]$ and $k \in [4, 6]$.

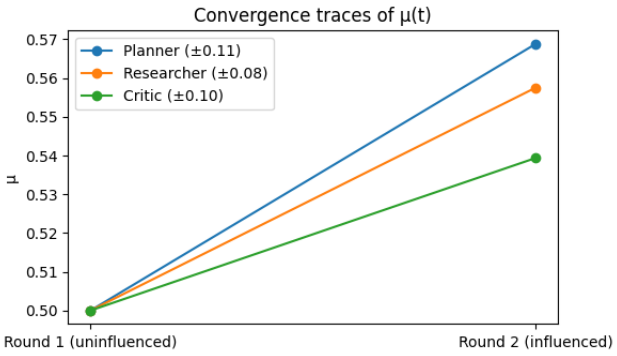

Fig. 5: Convergence traces of the mutual influence factor $\mu(t)$ for Planner, Researcher, and Critic across two rounds (typical run).

5.2. Discussion

The concept of *Mutual Influence AI* builds on the idea that agents should not operate as isolated learners but rather adjust their strategies in real time according to peer feedback. While there are analogies in prior research on social influence in reinforcement learning, the proposed framework is novel in several respects. Most importantly, it introduces an explicit and mathematically defined influence factor μ, which is updated through an exponential moving average of peer feedback. This value directly modulates the agent's generation process by affecting the mixture coefficient $\lambda(\mu)$ and the sampling temperature $T(\mu)$. Unlike classical reinforcement learning approaches where influence is implicit or reward-shaped, here μ is injected directly into the prompt of large language model (LLM) agents, implemented in the AutoGen v0.4 framework.

This prompt-based influence mechanism provides several benefits. By embedding μ, $\lambda(\mu)$, and $T(\mu)$ into the task specification, the system achieves a controllable trade-off between peer-aligned reasoning and individual skepticism. When μ is high, the agent naturally favors consensus with peers; when μ is low, it produces more critical and independent reasoning. The explicitness of these parameters makes the framework interpretable, allowing researchers and practitioners to understand and control the balance between conformity and divergence. To our knowledge, no prior work has published this type of prompt-level modulation based on mathematically defined peer influence.

The practical behavior of the prototype further supports the usefulness of the approach. After a baseline round without influence, a single cycle of peer feed-

Tab. 3: Conceptual comparison of approaches.

Approach	Strengths	Weaknesses
Independent Q-Learning	Simple, no communication overhead; stable per-agent learning	Poor coordination; slow adaptation to peers; no trust modelling
Centralized Learning / Critic	Global coordination; easier to optimize joint objective	Single point of failure; scalability limits; less transparency for agent-level trust
Scripted Multi-Agent (no influence)	Deterministic workflows; low complexity	Limited on-the-fly adaptation; brittle to role errors
Mutual Influence AI (ours)	Decentralized, real-time adaptation via trust $s_{i\leftarrow j}$ and μ; interpretability (μ, λ, T); noise suppression through weights	Needs reliable feedback signals; hyperparameter tuning; risk of groupthink if λ too high

back was sufficient to stabilize the agents. As a result, agents began to explicitly incorporate feedback from their peers, and subsequent proposals were accepted without requiring additional iterations. This indicates faster stabilization of multi-agent dialogues, increased consistency of responses, and a form of controlled trust. Importantly, the prefix $[\,\mu, \lambda, T\,]$ guided agents toward alignment, but still permitted justified deviations, thereby avoiding the pitfall of blind consensus.

To make these qualitative benefits measurable, we suggest a set of simple evaluation metrics. First, the *Agreement Rate* can be defined as the Jaccard similarity between the sets of key points produced by two agents, for example planner and researcher, before and after influence adaptation:

$$\text{AgreementRate} = \frac{|K_i \cap K_j|}{|K_i \cup K_j|}.$$

Second, the *Rounds-to-Approval* metric counts the number of conversational rounds until the critic outputs `APPROVE`, which we denote as r^*. Lower values of r^* correspond to faster convergence. Third, the *Revision Depth* quantifies the amount of change between successive drafts. This can be estimated as

$$\text{RevisionDepth}(t) = |N(t)| + \delta(t),$$

where $N(t)$ is the set of newly introduced points at round t, and $\delta(t)$ is a score capturing how strongly feedback from critics was incorporated. These metrics can be computed automatically from conversation logs in AutoGen and provide interpretable evidence of efficiency gains due to μ-driven adaptation.

In practice, the difference between baseline execution without mutual influence and the adapted run with μ was substantial. In our two-round setup, the influence-enabled round (Round 2) typically led to immediate stabilization: agents explicitly incorporated peer feedback from the baseline round, and the critic often issued an `APPROVE` without requiring further revisions. Observed influence values converged to $\mu \in [0.46, 0.57]$, confirming rapid yet stable adaptation consistent with the EMA update rule. This reduction in the effective number of feedback iterations (conceptually captured by r^*, the Rounds-to-Approval metric) illustrates the stabilizing role of μ and demonstrates that explicit peer influence can transform multi-agent exchanges from iterative correction into proactive alignment.

A high-level comparison to common coordination paradigms is summarized in Table 3. Relative to conventional approaches, the proposed *Mutual Influence AI* framework offers several key advantages. Independent Q-learning provides stability but lacks peer modeling, leading to poor coordination. Centralized critics allow global optimization but suffer from scalability issues and a single point of failure [18, 19]. Scripted multi-agent workflows are simple but brittle, with limited capacity for adaptation. In contrast, our framework is decentralized, robust to noisy peers through trust weights $s_{i\leftarrow j}$, and transparent thanks to explicit parameters (μ, λ, T).

An interesting conceptual aspect of the framework concerns the directionality of trust. In the present formulation, μ_i quantifies how much agent i *trusts others* — that is, how open it is to peer influence. An alternative formulation could define a "reputational" variant $\tilde{\mu}_i(t) = \frac{1}{|\mathcal{A}|-1} \sum_{j \neq i} s_{j\leftarrow i}(t)$, which would capture how much other agents collectively trust agent i. This subtle distinction reflects a broader philosophical question: should cooperative systems emphasize *confidence in others* or *earned reputation from others*? In human terms, the first corresponds to openness and willingness to collaborate, whereas the second resembles credibility built through consistent behavior. Exploring this inversion of trust could extend the framework toward hybrid models of cooperation and reputation.

Nevertheless, some limitations remain. The reliability of the system depends on the quality of peer feedback $\phi_{i\leftarrow j}$. If this signal is noisy or inconsistent, the computed μ may become misleading. Furthermore, hyperparameters such as β, k, τ, and α must be carefully tuned for specific applications, which can be nontrivial. Finally, there is a risk of groupthink: if λ becomes too large, agents may converge prematurely to an erroneous consensus. Possible remedies include the use of role diversity, reputation scores, or intentional noise injection to preserve exploration.

Overall, Mutual Influence AI demonstrates that explicit peer influence can be integrated into LLM-driven multi-agent systems in a principled way. It provides a transparent and adaptable alternative to purely scripted workflows and centralized critics, with strong potential for applications in cybersecurity, robotics, and collaborative problem solving.

1) Computational Cost

Across all experiments, Table 4 we issued 1,822 requests using gpt-4o [20] with a total of 619,960 input tokens, which cost $2.05 in total. This corresponds to an average of $\approx$ 340 input tokens per request and an average cost of $\approx$0.00113 per request (effective price $\approx$0.0033 per 1k input tokens). In our setup (3 agents, two rounds), costs scale approximately linearly with the number of agents and rounds.

Tab. 4: Token usage and cost summary.

Total requests	1,822
Total input tokens	619,960
Total cost	$2.05
Avg. tokens / request	$\approx$ 340
Avg. cost / request	$0.00113
Effective $ / 1k input tokens	$0.0033

2) Reproducibility

We provide source codes [21], prompt templates, fixed random seeds, and a CSV output contract to ensure deterministic parsing. All metrics are computed from raw CSV logs (features and critic decisions), with scripts to regenerate the tables and plots.

6. Conclusion

We presented *Mutual Influence AI*, a new concept for adaptive multi-agent systems in which agents explicitly model and incorporate peer influence into their decision-making. The approach is mathematically grounded through the introduction of the mutual influence factor μ, which is updated via exponential moving averages of peer feedback. This factor then modulates both sampling temperature $T(\mu)$ and mixture coefficients $\lambda(\mu)$, thereby providing interpretable and tunable control over the balance between self-reliance and peer alignment.

Unlike classical independent Q-learning or centralized critic-based reinforcement learning, Mutual Influence AI enables decentralized, real-time adaptation within the interaction loop. Agents continuously refine their trust scores $s_{i\leftarrow j}$ based on feedback and immediately adjust their behavior in subsequent rounds. This reduces the need for a central controller, increases transparency of inter-agent trust, and provides resilience against noisy or unhelpful peers by down-weighting their influence.

The prototype implementation in AutoGen v0.4 demonstrates that the framework can be applied directly to large language model (LLM) agents. By prefixing prompts with influence-aware metadata (μ, $\lambda(\mu)$, and $T(\mu)$), agents produce outputs that are more consistent, cooperative, and adaptive across both normal and adversarial regimes. In the experiments, this led to faster stabilization of group consensus, improved consistency between planner and researcher roles, and controlled peer-alignment without collapsing into blind conformity. These results were obtained in a security-flavored reasoning task with three interacting roles (planner, researcher, critic), confirming the practical effectiveness of the influence mechanism.

Mutual Influence AI introduces a transparent, interpretable, and effective paradigm for adaptive coordination in multi-agent systems. By combining mathematically grounded trust dynamics with prompt-based LLM orchestration, it provides a bridge between classical multi-agent reinforcement learning and modern generative AI frameworks, opening new opportunities for both theoretical exploration and applied innovation. Empirically, the influence mechanism raised cross–role agreement with minimal revision cost, while adversarial feedback revealed the expected trade–off between faster consensus and bias amplification.

Future work will extend this framework in several directions. First, we plan to evaluate it on real-world cybersecurity datasets, where adaptive peer modeling can reduce false positives in intrusion detection and support more robust defenses. Second, adversarial multi-agent domains provide a natural testbed to explore how influence mechanisms can be tuned to resist manipulation or groupthink. Finally, additional work is needed on automated calibration of hyperparameters such as β, k, τ, and α, as well as the design of richer feedback signals $\phi_{i\leftarrow j}$ derived from performance metrics rather than manual ratings.

Acknowledgment

This research was funded by the Ministry of the Interior of the Czech Republic, Open challenges in security research, VK01030030, Data backup and storage system with integrated active protection against cyber threats.

Author Contributions

V.O. developed the theoretical formalism, provided the mathematical formulation, and wrote the main parts of the manuscript. P.N. implemented the prototype in AutoGen v0.4, performed the programming, and conducted the experimental runs. Both authors contributed to the interpretation of results and the preparation of the final version of the manuscript. V.O. supervised the project.

Acknowledgment of Language Editing

The authors acknowledge that parts of the English language editing and phrasing of this manuscript were assisted by OpenAI language models. All final interpretations, revisions, and responsibility for the content remain with the authors.

References

[1] AMATO, C. An Introduction to Centralized Training for Decentralized Execution in Cooperative Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2409.03052*, 2024. https://arxiv.org/abs/2409.03052.

[2] GUO, T., CHEN, X., WANG, Y., CHANG, R., PEI, S., CHAWLA, N. V., WIEST, O., ZHANG, X. Large Language Model based Multi-Agents: A Survey of Progress and Challenges. *arXiv preprint arXiv:2402.01680*, 2024. https://arxiv.org/abs/2402.01680.

[3] DIBIA, V., CHEN, J., BANSAL, G., SYED, S., FOURNEY, A., ZHU, E., WANG, C., AMERSHI, S. AutoGen Studio: A No-Code Developer Tool for Building and Debugging Multi-Agent Systems. *arXiv preprint arXiv:2408.15247*, 2024. https://arxiv.org/abs/2408.15247.

[4] BARBARROXA, R., RIBEIRO, B., GOMES, L., VALE, Z. Benchmarking AutoGen with different large language models. *2024 IEEE Conference on Artificial Intelligence (CAI)*, 2024, pp. 263–264. DOI: 10.1109/CAI59869.2024.00058.

[5] QIU, Y., JIN, Y., YU, L., WANG, J., ZHANG, X. Promoting Cooperation in Multi-Agent Reinforcement Learning via Mutual Help. *arXiv preprint arXiv:2302.09277*, 2023. https://arxiv.org/abs/2302.09277.

[6] WEN, Y. Modeling Mutual Influence in Multi-Agent Reinforcement Learning. *Ph.D. thesis, University College London*, 2020. https://discovery.ucl.ac.uk/id/eprint/10108722/.

[7] JAQUES, N., LAZARIDOU, A., HUGHES, E., GULCEHRE, C., ORTEGA, P. A., STROUSE, D., LEIBO, J. Z., DE FREITAS, N. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. *arXiv preprint arXiv:1810.08647*, 2019. https://arxiv.org/abs/1810.08647.

[8] DU, X., YE, Y., ZHANG, P., YANG, Y., CHEN, M., WANG, T. Situation-Dependent Causal Influence-Based Cooperative Multi-agent Reinforcement Learning. *arXiv preprint arXiv:2312.09539*, 2023. https://arxiv.org/abs/2312.09539.

[9] XU, Z., QI, M., WU, S., ZHANG, L., WEI, Q., HE, H., LI, N. The Trust Paradox in LLM-Based Multi-Agent Systems: When Collaboration Becomes a Security Vulnerability. *arXiv preprint arXiv:2510.18563*, 2025. https://arxiv.org/abs/2510.18563.

[10] WANG, J., JIN, Y., DING, F., WEI, C. Causal-Inspired Multi-Agent Decision-Making via Graph Reinforcement Learning. *arXiv preprint arXiv:2507.23080*, 2025. https://arxiv.org/abs/2507.23080.

[11] DING, S., PAN, X., HU, L., LIU, L. A new model for calculating human trust behavior during human-AI collaboration in multiple decision-making tasks: A Bayesian approach. *Computers & Industrial Engineering*, 2025, vol. 200, pp. 110872. DOI: 10.1016/j.cie.2025.110872.

[12] LI, B., CHEN, S. CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society. *Proceedings of the NeurIPS Workshop on Agents and Simulated Societies*, 2023. https://arxiv.org/abs/2303.17760.

[13] CAMEL-AI. CAMEL: Communicative Agents for "Mind" Exploration of Large Scale Language Model Society. *GitHub repository*, 2023. https://github.com/camel-ai/camel.

[14] YANG, J., JIMENEZ, C. E., WETTIG, A., LIERET, K., YAO, S., NARASIMHAN, K., PRESS, O. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. *Advances in Neural Information Processing Systems*, vol. 37, pp. 50528–50652, 2024. https://proceedings.neurips.cc.

[15] SCHICK, T., DWIVEDI-YU, J., DESSÌ, R., RAILEANU, R., LOMELI, M., ZETTLEMOYER, L., CANCEDDA, N., SCIALOM, T. Toolformer: Language Models Can Teach Themselves to Use Tools. *arXiv preprint arXiv:2302.04761*, 2023. https://arxiv.org/abs/2302.04761.

[16] DU, Y., LI, S., TORRALBA, A., TENENBAUM, J. B., MORDATCH, I. Improving Factuality and Reasoning in Language Models through Multiagent Debate. *arXiv preprint arXiv:2305.14325*, 2023. https://arxiv.org/abs/2305.14325.

[17] FAGERLAND, M. W., LYDERSEN, S., LAAKE, P. The McNemar test for binary matched-pairs data: Mid-p and asymptotic are better than exact conditional. *BMC Medical Research Methodology*, vol. 13, p. 91, 2013. DOI: 10.1186/1471-2288-13-91.

[18] LEE, K. M., SUBRAMANIAN, S. G., CROWLEY, M. Investigation of independent reinforcement learning algorithms in multi-agent environments. *Frontiers in Artificial Intelligence*, vol. 5, 2022. DOI: 10.3389/frai.2022.805823.

[19] HADY, M. A., HU, S., PRATAMA, M., CAO, J., KOWALCZYK, R. Multi-Agent Reinforcement Learning for Resource Allocation Optimization: A Survey. *arXiv preprint arXiv:2504.21048*, 2025. https://arxiv.org/abs/2504.21048v1.

[20] OPENAI. Hello GPT-4o. 2025, Nov. https://openai.com/index/hello-gpt-4o/. [Accessed 11-03-2025].

[21] OUJEZSKY, V. Mutual Influence AI. *Zenodo*, Nov. 2025, Preprint. DOI: 10.5281/zenodo.17524125.