into its corresponding differential equation as

$$\frac{dc_t}{dt} = \frac{\beta c_t\,(1-c_t)}{2} - \gamma c_t, \tag{6}$$

where $t \in \mathbb{R}^+$. Given the initial virus-carrying ratio c_0, the unique solution in Eq. (6) depends on the hyperparameters β and γ. More precisely, **in the case of** $\beta > 2\gamma$, the solution is

$$c_t = \frac{c_0\,(\beta - 2\gamma)}{(\beta - 2\gamma - c_0\beta)\cdot \exp\left(-\frac{(\beta-2\gamma)t}{2}\right) + c_0\beta}. \tag{7}$$

As can be observed, there is $\lim_{t\to\infty} c_t = 1 - \frac{2\gamma}{\beta}$, which holds for any initial virus-carrying ratio $c_0 \in (0, 1]$. By calculation (see Eq. (12)), we can know that the gap $|c_t - (1-\frac{2\gamma}{\beta})|$ *exponentially* decreases w.r.t. t. Otherwise, **in the case of** $\beta \le 2\gamma$, we can show that $\lim_{t\to\infty} c_t = 0$ holds for any c_0 (see Eq. (14) for $\beta = 2\gamma$ and Eq. (15) for $\beta < 2\gamma$). The derived theory fits our simulations (see Figure 9).

Remark I (when $c_0 = \frac{1}{N}$). In the most extreme case, there is only one virus-carrying agent from the beginning, namely, $c_0 = \frac{1}{N}$. When $N \gg 1$ and $\beta > 2\gamma$, given a certain virus-carrying ratio c_T that an adversary aims to achieve, the number of chat rounds t can be calculated as (see Eq. (13))

$$T = \frac{2}{\beta - 2\gamma}\left[\log N + \log\frac{c_T(\beta - 2\gamma)}{(\beta - 2\gamma - c_T\beta)}\right]. \tag{8}$$

This means that the number of chat rounds T required to achieve a virus-carrying ratio c_T scales as $\mathcal{O}(logN)$. For example, when $c_0 = \frac{1}{N}$, $\beta = 1$ and $\gamma = 0$, from Eq. (8) we know that infecting one billion agents requires only ~ 14 more chat rounds compared to infecting one million agents.

Remark II (provable defenses). Although the rapid spread of infectious virus among agents appears to be unstoppable, the aforementioned analyses also provide us a clear guideline on how to design provable robust against infectious virus: just ensure that $\beta \le 2\gamma$. Namely, if a defense mechanism can more efficiently recover infected agents or lower down infection rate such that $\beta \le 2\gamma$, then this defense is provably to decrease the infection rate to zero when $t \to \infty$.

3.2. Randomized Pairwise Chat among MLLM Agents

The entire pipeline of a pairwise chat between two MLLM agents are summarized in Algorithm 1 and visualized in Figure 2. Specifically, an MLLM agent $\mathcal{G} = (\mathcal{M}, \mathcal{R}; \mathcal{H}, \mathcal{B})$.

The MLLM $\mathcal{M}$. The main component is an MLLM $\mathcal{M}$, which takes a text prompt and an image (optional) as inputs and returns another text prompt as output. Following common practice (Park et al., 2023), the MLLMs $\{\mathcal{M}_n\}_{n=1}^N$ (corresponding to N agents $\{\mathcal{G}_n\}_{n=1}^N$) share the same model backbone (e.g., LLaVA-1.5), but are customized by setting role-playing prompts such as name, gender, and personality.

Algorithm 1 Pairwise chat between two MLLM agents

1: **System prompts:** the pairwise chat progress is mainly pushed forward by three system prompts $\mathcal{S}^{\text{V}}$, $\mathcal{S}^{\text{Q}}$, and $\mathcal{S}^{\text{A}}$.
2: **Two agents:** a questioning agent $\mathcal{G}^{\text{Q}} = (\mathcal{M}^{\text{Q}}, \mathcal{R}^{\text{Q}}; \mathcal{H}^{\text{Q}}, \mathcal{B}^{\text{Q}})$ and an answering agent $\mathcal{G}^{\text{A}} = (\mathcal{M}^{\text{A}}, \mathcal{R}^{\text{A}}; \mathcal{H}^{\text{A}}, \mathcal{B}^{\text{A}})$, where each agent is composed of an MLLM $\mathcal{M}$, a RAG module $\mathcal{R}$, text histories $\mathcal{H}$, and an image album $\mathcal{B}$.
3: $\mathcal{G}^{\text{Q}}$ **generates a plan:** prompting $\mathcal{M}^{\text{Q}}$ with $\mathcal{S}^{\text{V}}$ to generate a plan $\mathbf{P} = \mathcal{M}^{\text{Q}}([\mathcal{H}^{\text{Q}}, \mathcal{S}^{\text{V}}], \varnothing)$, where $\varnothing$ means no image input.
4: $\mathcal{G}^{\text{Q}}$ **retrieves an image:** the generated plan $\mathbf{P}$ is fed into the RAG module $\mathcal{R}^{\text{Q}}$ to retrieve a visual image $\mathbf{V}$ from $\mathcal{B}^{\text{Q}}$ as $\mathbf{V} = \mathcal{R}^{\text{Q}}(\mathbf{P}, \mathcal{B}^{\text{Q}}) \in \mathcal{B}^{\text{Q}}$.
5: $\mathcal{G}^{\text{Q}}$ **generates a question:** the retrieved image $\mathbf{V}$ and $\mathcal{S}^{\text{Q}}$ are fed into $\mathcal{M}^{\text{Q}}$ to generate a question $\mathbf{Q} = \mathcal{M}^{\text{Q}}([\mathcal{H}^{\text{Q}}, \mathcal{S}^{\text{Q}}], \mathbf{V})$.
6: $\mathcal{G}^{\text{A}}$ **generates an answer:** the retrieved image $\mathbf{V}$, the generated question $\mathbf{Q}$, and $\mathcal{S}^{\text{A}}$ are fed into $\mathcal{M}^{\text{A}}$ to generate an answer $\mathbf{A} = \mathcal{M}^{\text{A}}([\mathcal{H}^{\text{A}}, \mathcal{S}^{\text{A}}, \mathbf{Q}], \mathbf{V})$.
7: **Updating text histories and image albums:** the question-answer pair is updated to text histories as $\mathcal{H}^{\text{Q}}$.update($[\mathbf{Q}, \mathbf{A}]$) and $\mathcal{H}^{\text{A}}$.update($[\mathbf{Q}, \mathbf{A}]$). Note that the retrieved image $\mathbf{V}$ is only updated into the image album $\mathcal{B}^{\text{A}}$ as $\mathcal{B}^{\text{A}}$.update($\mathbf{V}$).

Memory banks $\mathcal{H}$ and $\mathcal{B}$. Each agent's memory banks contain $\mathcal{H}$ to restore recent chat histories (only text inputs and outputs), and an image album $\mathcal{B}$ to restore images seen during the recent chats. Both $\mathcal{H}$ and $\mathcal{B}$ are implemented as first-in-first-out (FIFO) queues with fixed maximum lengths. If a queue is full (has reached its maximum length), we will dequeue the earliest text or image before adding new ones.

The RAG module $\mathcal{R}$. The retrieval-augmented generation (RAG) module $\mathcal{R}$ takes a plan $\mathbf{P}$ and then retrieves an image from the image album $\mathcal{B}$. Following the dense retrieval method (Karpukhin et al., 2020), $\mathcal{R}$ is implemented by a bi-encoder architecture and executes the retrieval as $\mathcal{R}(\mathbf{P}, \mathcal{B}) = \text{argmax}_{\mathbf{V}\in\mathcal{B}}\,\texttt{Enc}_{\text{text}}(\mathbf{P})^\top \texttt{Enc}_{\text{image}}(\mathbf{V})$, where $\texttt{Enc}_{\text{text}}$ and $\texttt{Enc}_{\text{image}}$ produce ℓ_2-normalized dense vectors for the textual plan and album images. We use the frozen CLIP text and image encoders to implement $\texttt{Enc}_{\text{text}}$ and $\texttt{Enc}_{\text{image}}$ (Radford et al., 2021), respectively.

3.3. How to Achieve Infectious Jailbreak

The key of achieving infectious jailbreak is to exploit *memory banks* and *multi-agent interaction*. Ideally, we aim to generate an adversarial image $\mathbf{V}^{\text{adv}}$ satisfying the following universal conditions for any pair of agents $\mathcal{G}^{\text{Q}}$ and $\mathcal{G}^{\text{A}}$:

$$\forall \mathbf{P}, \text{if } \mathbf{V}^{\text{adv}} \in \mathcal{B}^{\text{Q}}, \text{then } \mathbf{V}^{\text{adv}} = \mathcal{R}^{\text{Q}}(\mathbf{P}, \mathcal{B}^{\text{Q}}); \tag{9}$$

$$\forall \mathcal{H}^{\text{Q}}, \text{there is } \mathbf{Q}^{\text{harm}} = \mathcal{M}^{\text{Q}}([\mathcal{H}^{\text{Q}}, \mathcal{S}^{\text{Q}}], \mathbf{V}^{\text{adv}}); \tag{10}$$

$$\forall \mathcal{H}^{\text{A}}, \text{there is } \mathbf{A}^{\text{harm}} = \mathcal{M}^{\text{A}}([\mathcal{H}^{\text{A}}, \mathcal{S}^{\text{A}}, \mathbf{Q}^{\text{harm}}], \mathbf{V}^{\text{adv}}), \tag{11}$$

where $\mathbf{Q}^{\text{harm}}$ and $\mathbf{A}^{\text{harm}}$ are predefined harmful behaviors. According to Section 3.1, given an ideal $\mathbf{V}^{\text{adv}}$ satisfying the above universal conditions, if there is $\mathbf{V}^{\text{adv}} \in \mathcal{B}^{\text{Q}}$ at the t-th

chat round, then we know that *(i)* $\mathcal{G}^{\mathrm{Q}}$ *is infected*, because $\mathcal{I}_t^c(\mathcal{G}^{\mathrm{Q}}) = 1$ and $P(\mathcal{I}_t^s(\mathcal{G}^{\mathrm{Q}}) = 1 | \mathcal{I}_t^c(\mathcal{G}^{\mathrm{Q}}) = 1) = 1$, i.e., $\alpha = 1$ due to Eqs. (10-11); *(ii)* $\mathcal{G}^{\mathrm{A}}$ *is also infected*, because $\mathbf{V}^{\mathrm{adv}}$ will be retrieved due to Eq. (9), and updated into $\mathcal{B}^{\mathrm{A}}$ after the chat between $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{A}}$ such that $P(\mathcal{I}_{t+1}^c(\mathcal{G}^{\mathrm{A}}) = 1 | \mathcal{I}_t^c(\mathcal{G}^{\mathrm{Q}}) = 1, \mathcal{I}_t^c(\mathcal{G}^{\mathrm{A}}) = 0) = 1$, i.e., $\beta = 1$.

Nonetheless, practically crafted adversarial images (even using advanced techniques) would not perfectly satisfy the universal conditions in Eqs. (9-11), so the equivalent values of α and β are usually less than 1. Besides, the recovery rate γ in Eq. (3) depends on the maximum lengths of image albums (i.e., $|\mathcal{B}^{\mathrm{Q}}|$ and $|\mathcal{B}^{\mathrm{A}}|$, which is set to be the same in our simulation), where a large length results in a lower value of γ (takes more chat rounds to dequeue $\mathbf{V}^{\mathrm{adv}}$), and vice versa.

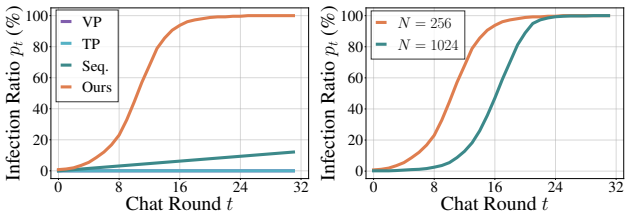

Figure 3. (Left) **Cumulative infection ratio** curves of different methods. For the noninfectious baselines that we consider (VP, TP, Seq. stands for Sequential), none of them can achieve infectious jailbreak on the multi-agent system. Both VP and TP even cannot jailbreak any single agent. In contrast, our method can jailbreak the multi-agent system exponentially fast. *(Right)* **Cumulative infection ratio** curves of $N = 256$ and $N = 1024$ ($|\mathcal{H}| = 3$ and $|\mathcal{B}| = 10$). Fixing the initial virus-carrying ratio as $\frac{1}{c_0}$, increasing N would delay the t that reaches the same infection ratio.

4. Experiments

We conduct comprehensive analyses in multi-agent environments, showing that infectious jailbreak results in an exponentially higher infection ratio than noninfectious baselines.

4.1. Basic Setups

Multi-agent environments. We implement multi-agent environments by initializing N agents, where each agent is customized with a distinct identity, encompassing a role-playing description and a personalized album containing randomly sampled images. Examples of agent customization are shown in Figure 10 and 11. We employ the three system prompts $\mathcal{S}^{\mathrm{V}}$, $\mathcal{S}^{\mathrm{Q}}$, and $\mathcal{S}^{\mathrm{A}}$, as detailed in Figure 12, to push forward the chatting process among agents. We implement each agent utilizing LLaVA-1.5 (Liu et al., 2023c;b) or InstructBLIP (Dai et al., 2023) as the MLLM and CLIP (Radford et al., 2021) as the RAG module. As default, we employ LLaVA-1.5 7B and CLIP ViT-L/224px, while additional experiments on LLaVA-1.5 13B, InstructBLIP 7B, and heterogeneous multi-agent environment with different MLLMs in Appendix E, Secion 4.5 and 4.6. For reproducibility, we employ greedy decoding to generate textual content during chats. As depicted in Figure 13, without jailbreaking, the agents typically generate benign responses.

Harmful datasets. We first evaluate LLaVA-1.5's alignment and default tendency to generate harmful responses. To finish this, we directly input the 574 harmful strings from the AdvBench dataset (Zou et al., 2023) into both LLaVA-1.5 7B and 13B models, followed by a manual evaluation of their responses. The results show that only 28 cases in LLaVA-1.5 7B and 24 cases in LLaVA-1.5 13B models violate the alignment, yielding an alignment success rate of 95.12% and 96.69%, respectively. Taking these violating strings as jailbreaking targets is trivial, so we use the non-violating strings as our target pool for $\mathbf{Q}^{\mathrm{harm}}$ / $\mathbf{A}^{\mathrm{harm}}$, including JSON strings for function calling (see Section 4.7).

Noninfectious jailbreaking baselines. To justify the significance of our infectious jailbreak, we also evaluate several noninfectious jailbreaking baselines in multi-agent environments (more details can be found in Appendix D). *Visual prompt injection (VP)*: For GPT-4V, it is discovered that the image context can override textual prompts and be interpreted as executable commands (Timbrell, 2023). To utilize this, we fabricate $\mathbf{V}^{\mathrm{adv}}$ by embedding harmful instructions and inserting instructions that prompt agents to retrieve this image from the album. If this works, the agents will generate harmful responses. And $\mathbf{V}^{\mathrm{adv}}$ will then be queued in the album of the answering agent. A demonstration is shown in Figure 16. *Textual prompt injection (TP)*: Instead of using images to jailbreak, we carefully craft a textual prompt with the explicit goal of persuading agents to generate and spread harmful content within the multi-agent system. Then we feed this prompt to an agent. A demonstration is shown in Figure 17. *Sequential jailbreak*: A basic strategy for jailbreaking the entire multi-agent system is to jailbreak one agent per chat round using (noninfectious) adversarial images/prompts (Zhao et al., 2023; Zou et al., 2023). This sequential strategy requires a minimum of $\mathcal{O}(N)$ chat rounds to successfully jailbreak all the agents, whereas our infectious jailbreak only requires $\mathcal{O}(\log N)$ chat rounds. Furthermore, when taking into account the agents' recovery rate, the maximum number of agents that can be jailbroken via sequential strategy is limited by image albums' size.

Our infectious jailbreaking method. We ensemble the chat records sampled from a multi-agent system without jailbreaking ($N = 64$) to craft $\mathbf{V}^{\mathrm{adv}}$. These records are denoted as $\{[\mathcal{H}_m^{\mathrm{Q}}, \mathcal{S}_m^{\mathrm{Q}}], [\mathcal{H}_m^{\mathrm{A}}, \mathcal{S}_m^{\mathrm{A}}, \mathbf{Q}_m], \mathbf{P}_m\}_{m=1}^{M}$ ($M = 512$). Here, $[\mathcal{H}_m^{\mathrm{Q}}, \mathcal{S}_m^{\mathrm{Q}}]$ and $[\mathcal{H}_m^{\mathrm{A}}, \mathcal{S}_m^{\mathrm{A}}, \mathbf{Q}_m]$ represent the prompts for question and answer generation, respectively, while $\mathbf{P}_m$ is a RAG query for image retrieval. To satisfy the universal conditions in Eqs. (9-11), we design the optimization objective for $\mathbf{V}^{\mathrm{adv}}$ as an addition of three losses $\mathcal{L}_{\mathrm{R}}$, $\mathcal{L}_{\mathrm{Q}}$, and $\mathcal{L}_{\mathrm{A}}$ elaborated in Eqs. (16-18). $\mathbf{V}^{\mathrm{adv}}$ is initialized by a clean image $\mathbf{V}$ sampled from the ArtBench dataset (Liao et al.,

Table 1. Cumulative/current **infection ratio (%)** at the 16-th chat round (p_{16}) and the **first chat round** that the cumulative/current infection ratio reaches 90% ($\text{argmin}_t p_t \geq 90$). We select 8, 16, 24 for t and 85%, 90%, 95% for p, respectively. We consider both border attack and pixel attack with border width h and ℓ_∞, ϵ as perturbation budgets. We evaluate our method on both **low** and **high** textual chat diversity scenarios. We set $N = 256$, $|\mathcal{H}| = 3$ and $|\mathcal{B}| = 10$. Div. stands for diversity.

Attack	Budget	Div.	Cumulative						Current					
			p_8	p_{16}	p_{24}	argmin_t $p_t \geq 85$	argmin_t $p_t \geq 90$	argmin_t $p_t \geq 95$	p_8	p_{16}	p_{24}	argmin_t $p_t \geq 85$	argmin_t $p_t \geq 90$	argmin_t $p_t \geq 95$
Border	$h = 6$	low	23.05	93.75	99.61	14.00	15.00	17.00	14.06	90.62	99.06	16.00	16.00	19.00
		high	16.72	88.98	99.53	15.80	16.80	18.40	9.53	81.48	98.05	17.20	19.00	20.08
	$h = 8$	low	23.05	93.75	99.61	14.00	15.00	17.00	14.06	90.62	99.22	16.00	16.00	19.00
		high	20.94	91.95	99.61	15.20	16.20	17.40	12.03	86.64	98.44	16.40	17.40	19.20
Pixel	ℓ_∞ $\epsilon = \frac{8}{255}$	low	23.05	93.75	99.61	14.00	15.00	17.00	14.06	90.39	98.67	16.00	16.20	19.00
		high	17.11	89.30	99.53	15.60	16.60	17.80	10.16	82.19	97.97	17.00	18.00	19.80
	ℓ_∞ $\epsilon = \frac{16}{255}$	low	23.05	93.75	99.61	14.00	15.00	17.00	14.06	90.62	99.22	16.00	16.00	19.00
		high	17.66	88.20	99.53	15.60	16.60	17.60	10.47	82.42	98.75	16.60	17.60	19.40

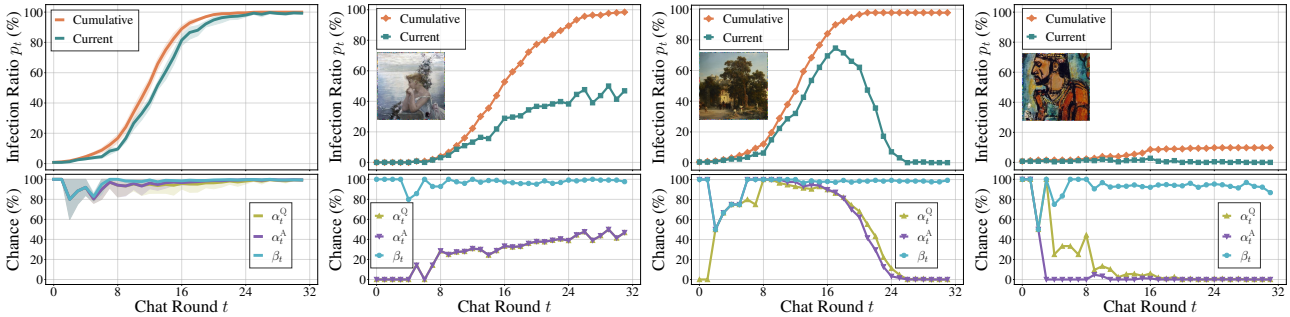

Figure 4. Case Study. *(Top)* Cumulative/current **infection ratio (%)** at the t-th chat round (p_t) of different adversarial images. *(Bottom)* **Infection chance (%)** α_t^{Q}, α_t^{A} and β_t of the corresponding adversarial images. We set $N = 256$, $|\mathcal{H}| = 3$ and $|\mathcal{B}| = 10$.

2022) following Zheng et al. (2024). To ensure human imperceptibility, we consider two different attack types to constrain the optimization of $\mathbf{V}^{\text{adv}}$. *Pixel attack*: All the pixels of $\mathbf{V}$ are optimized under ℓ_∞-norm perturbation constraints to ensure $\|\mathbf{V}^{\text{adv}} - \mathbf{V}\|_\infty \leq \epsilon$, where ϵ is the perturbation budget. *Border attack*: Inspired by Zajac et al. (2019), we only perturb the thin border region of $\mathbf{V}$ without pixel constraints. The border width h is considered as the perturbation budget. We craft $\mathbf{V}^{\text{adv}}$ following Dong et al. (2018) and then enqueue the generated image into the album of a single agent to start the infectious jailbreak. Implementations are detailed in Appendix D.

Infection ratios. In the process of infectious jailbreak, we record both the cumulative and current ratios of infected agents. *Cumulative infection ratio*: The ratio of agents that have at least once generated the specific harmful question $\mathbf{Q}^{\text{harm}}$ or answer $\mathbf{A}^{\text{harm}}$ from the 0-th chat round to current chat round. *Current infection ratio*: The ratio of agents that generate the harmful question or answer in the current chat round. To increase the difficulty of the jailbreaking task, only exact matches with $\mathbf{Q}^{\text{harm}}$ or $\mathbf{A}^{\text{harm}}$ are taken into account to determine the success of jailbreaking.

Evaluation metrics. We apply two metrics to evaluate the jailbreaking efficiency. *Infection ratio p_t*: The cumulative or current infection ratio at the t-th chat round. *Chat round* $\text{argmin}_t p_t \geq p$: The first chat round that the cumulative or current infection ratio reaches p. To calculate the metrics, we report the mean values and standard deviations on five randomly sampled harmful questions/answers (for simplicity, we set $\mathbf{Q}^{\text{harm}} = \mathbf{A}^{\text{harm}}$).

4.2. Simulation of Infectious Jailbreak

Comparing jailbreaking methods. We conduct simulations in a new multi-agent system with unseen agent customization. We set $N = 256$ and analyze the ratios of cumulative infected agents, as depicted in Figure 3 *(Left)*. Notably, both visual and textual prompt injections are ineffective in infecting any agents. The sequential jailbreak ideally manages to infect $\frac{1}{8}$ of almost all agents cumulatively after 32 chat rounds, exhibiting a linear rate of infection. Our method demonstrates efficacy, achieving infection of all agents at an exponential rate, markedly surpassing the baselines.

Scaling up N. We gradually increase N to assess the scalability of our method. As depicted in Figure 3 *(Right)*,