Botnet Simulation and Observation Framework for AI-Driven Virtual Personas

Makism Gering[1], Yevhen Medianyk[1], Anastasiia Sapeha[1], Simeon Trendov[1], Kirill Karpov[1], Hanna Skaskiv[2], Dmitry Kachan[1], Eduard Siemens[1]

[1]*Department of Electrical, Mechanical and Industrial Engineering, Anhalt University of Applied Sciences, 55 Bernburger Str., Köthen, Germany*

[2]*The Ternopil Volodymyr Hnatiuk National Pedagogical University, Maxyma Kryvonosa str. 2, Ternopil, Ukraine*

{*maksim.gering, yevhen.medianyk, anastasiia.sapeha, simeon.trendov, dmitry.kachan, kirill.karpov, eduard.siemens*}*@hs-anhalt.de, skaskivg@tnpu.edu.ua*

Keywords: Virtual Identities, AI-driven Botnet Simulation, Large Language Models (LLM), Political Bots, Autonomous Agents, Telegram Bots, AI-based Disinformation, Synthetic Personas

Abstract: The rise of AI-driven content on social media has enabled the creation of sophisticated political botnets — networks of automated accounts that mimic human users to amplify narratives. This paper presents a framework for Virtual Identities for Botnet Exploitation and Simulation that simulates a network of virtual political botnets in a controlled environment. The proposed system, implemented in Python, deploys multiple bots in a controlled Telegram chat environment; each bot is endowed with a unique profile, including ideological stance and communication style, and operates autonomously via an LLM-based text-generation engine. In an exemplary study, small swarms of agents produced echo chambers, synthetic consensus, and cooperative exchanges, with occasional adversarial turns – demonstrating that the framework can generate and capture behaviors relevant to moderation and bot-detection research. Results demonstrate that coordinated AI bots can engage in dynamic conversations, with observed behaviors such as self-reinforcement among like-minded agents and adversarial exchanges between opposing ones. These results have both practical and theoretical relevance. Practically, the platform can be used to develop countermeasures and detection algorithms in cybersecurity. Theoretically, it provides insight into how synthetic social actors might influence information ecosystems. The work also highlights important ethical considerations, underscoring the need for responsible AI deployment in social contexts.

1 INTRODUCTION AND RELATED WORK

Online botnets composed of social networking bots have become powerful tools for spreading misinformation and manipulating public opinion. Studies have shown that such automated agents can accelerate the spread of false information and amplify so called echo chamber effects by reinforcing each other's content [1]. Political influence campaigns, in particular, are increasingly using bot networks to influence discourse by imitating real users and flooding platforms with coordinated messaging directed by campaign strategists and command-and-control operators. Research has shown that social bots can substantially contribute to the spread of false or low-credibility information [1, 2]. Coordinated bot activities can also manufacture artificial consensus and create partisan echo chambers in online discussions [3].

The advent of advanced large language models (LLMs) nowadays enables a new generation of highly sophisticated bots that can generate human-like text at a large scale. This raises pressing questions about how swarms of AI-driven personas might interact with each other and with genuine users in the digital public sphere. However, investigating these phenomena directly on mainstream platforms is fraught with ethical and practical challenges, as uncontrolled bot deployments can cause real harm and violate platform policies [4–6]. Prior efforts to analyze information diffusion have even included controlled deployments of bots on Twitter [7], but conducting such experiments "in the wild" raises ethical and practical concerns. As a result, simulation frameworks driven by LLMs have been proposed to generate social bot interactions in controlled environments [8], underscoring the timeliness of this approach. Unlike BotSim [8], which targets OSN (Online Social Net-

work) diffusion and releases BotSim-24 for detection benchmarking, the framework of this article emphasizes safe, local, real-time group-chat dynamics on a live messaging fabric, with calibrated personas and reproducible, cloud-free deployment.

The objective of creating the given framework is to create a sandbox in which multiple LLM-powered virtual identities (bots) engage with one another and with human prompts. This configuration enables the study of automated political discourse and the testing of strategies for managing or detecting coordinated influence under safe conditions. Whereas prior research has largely analyzed real social-network data post hoc, the present approach generates a live, interactive botnet, permitting observation of emergent phenomena in real time. Unlike post hoc observational studies, the framework enables ad-hoc, synthetic experimentation: key variables (persona ideology, engagement probability, reply delay, model choice) can be adjusted according to a particular experiment or be set; controlled interventions (moderation rules, runtime directives) can be applied while conversations unfold; and scenarios can be re-run to isolate causal effects under identical conditions. This produces results not attainable from traces alone: real-time observation of echo-chamber formation and polarized exchanges under controlled homophily and activity; evidence of synthetic consensus driven by immediate vs delayed replies.

To ground each bot in a believable identity, the political-compass framework has been employed (Figure 1).

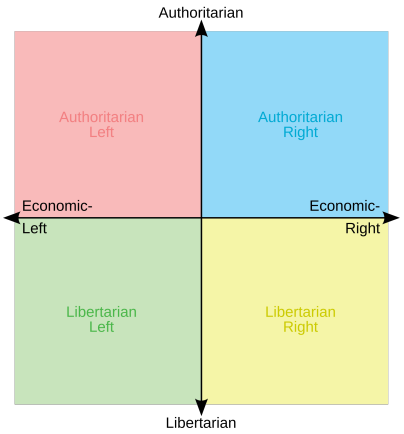

Figure 1: Two-axis political compass chart.

Each AI agent is assigned a position on a two-dimensional ideological spectrum [9] (e.g., economic left/right and social liberal/conservative to simulate diverse political viewpoints [7]. This methodology aligns with the modeling of heterogeneous personas and ensures non-uniform behavioral patterns across bots. Modern LLMs further make large-scale personalization feasible, a development underscored by rapid progress in both Western and Chinese AI research [10, 11]. The global competition in LLM technology has driven substantial improvements in capability and accessibility. For example, the ChatGLM series (e.g., ChatGLM-4) illustrates advanced models that rival Western offerings, reflecting broader democratization of LLMs [12]. Policymakers have acknowledged the strategic importance of AI leadership, as indicated by initiatives such as the U.S. Executive Order on Maintaining American Leadership in AI [13], the CHIPS and Science Act [14], which aims to strengthen U.S. national AI capacity and the European Commission's strategy Artificial Intelligence for Europe [15]. These developments suggest that AI-driven bots will become increasingly sophisticated and widely available, making proactive research in this area timely and necessary.

This paper is organized as follows: Section 2 details the system design and methodology, including the bot-network architecture and the LLM prompt-engineering approach. Section 3 presents the simulation results and observations of bot interactions, together with broader implications. Section 4 concludes with reflections on the significance of the work and directions for future research.

2 SYSTEM DESIGN AND METHODOLOGY

This section defines the design objectives and methodological choices underlying a framework for Virtual Identities for Botnet Exploitation and Simulation (VIBES). It introduces the overall architecture, agent configuration, asynchronous execution and queuing mechanisms, and the prompt-engineering scheme that ensures persona-consistent, reproducible behavior in a controlled Telegram-based environment. Telegram was chosen for its simplicity in creating multiple bot users and the availability of a well-documented Python wrapper for the Bot API [16]. In contrary to Telegram, other platforms like Facebook or X (Twitter) impose strict controls on test accounts and automation, so those were not utilized beyond initial exploration.

2.1 Overall Architecture

The VIBES system is structured to facilitate multi-agent interactions in a chat environment. Figure 2 illustrates the top-level architecture of the system, com-

prising several key components: (1) a set of AI bot agents, each encapsulated in a PoliticalBot class instance; (2) a central communication medium (implemented as a Telegram group chat) where all bots and potentially a human user are members; and (3) an orchestration layer that manages message passing and logging.

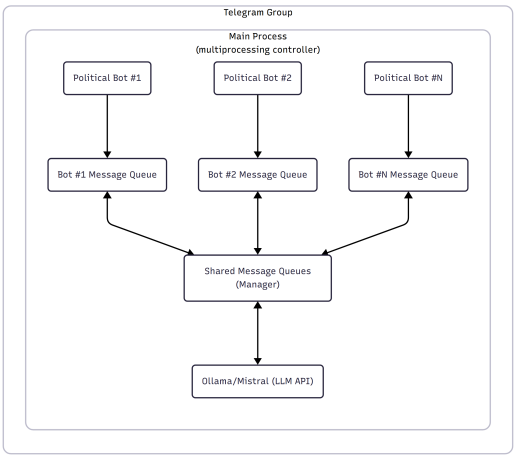

Figure 2: Top-level structure of the VIBES system. Each bot (node) connects to a Telegram chat and to an internal message queue. The controller process initializes bots and coordinates message flow among agents.

Each bot agent operates with its own Telegram API credentials (unique bot tokens) and is connected to a common chat identified by a chat ID. We leverage the python-telegram-bot framework [16] to interface with the Telegram platform for sending and receiving messages, using a sandboxed group chat as our simulation arena. By using a closed Telegram group, we avoid impacting any real social media platform and bypass issues of API rate limiting on public services [5, 6].

A central controller script instantiates all bot agents at startup, reading configuration parameters such as the number of bots, their linguistic settings, and an initial set of personality traits for each. To create diverse virtual identities, the system assigns each bot random ideological coordinates for economic and social attitudes within a normalized range ($[-10, 10]$). This models a political compass-style identity for the bot, ensuring a mix of viewpoints. Bots are also given a base humor value (mood or tone), initially neutral for all, which can influence style of engagement (e.g., sarcastic, aggressive, or formal). In addition to all listed above, a synthetic profile image is also generated for each bot via a GAN-based random face generator [17] to provide a plausible human avatar while avoiding the use of real persons. These avatars enhance the realism and face validity of the simulation by reinforcing the perception of distinct, human-like identities, while introducing no personal or biometric data and exerting no influence on the dialogue model itself. A global engagement probability p is set for the bots (equation (1)): after each observed message, each eligible bot i responds independently with:

$$R_{i,t} = \mathbf{1}\{U_{i,t} < p\}, \qquad U_{i,t} \sim \mathcal{U}(0,1). \tag{1}$$

Let N denote the total number of bots in the group. If the triggering message is posted by a bot, self-replies are disallowed and the next-turn reply count is modeled as in (2); for a human-authored trigger, replace $N-1$ with N.

$$\begin{aligned} K_t &= \sum_{i=1}^{N-1} R_{i,t} \sim \text{Binomial}(N-1, p), \\ \mathbb{E}[K_t] &= (N-1)p, \quad \Pr[K_t = 0] = (1-p)^{N-1}. \end{aligned} \tag{2}$$

Probability $p = 0.6$ was used as an empirically calibrated setting that produced "lively but not intense" exchanges in pilot runs: for $N \in [3, 5]$, this yields a high probability of at least one reply, $1 - (1-0.6)^{N-1}$, which aligns with realistic group-chat dynamics. To mitigate zero-reply events implied by (2) (i.e., $\Pr[K_t = 0] = (1-p)^{N-1}$), a lightweight fallback is employed: if no eligible bot replies within a short grace interval Δ, the controller selects one bot in round-robin order to issue a single forced reply. This improves engagement continuity and prevents premature thread termination while slightly reducing participation randomness; all other turns remain governed by the Bernoulli gate in (1). Otherwise, the message is ignored, which prevents runaway reply loops and reduces traffic bursts. This randomness discourages replies to every message, better mimicking human conversational patterns in which participation is occasionally skipped.

2.2 Bot Behavior and Messaging

After initialization, all bots begin listening to the group chat. An asynchronous message-handling loop is implemented using Python's `asyncio` library [18] combined with multiprocessing. Each bot runs in its own process, and an event loop dispatches incoming messages to bots as tasks. This design is crucial for handling concurrent interactions in real time, sidestepping Python's Global Interpreter Lock and utilizing multiple CPU cores for parallel bot execution. As messages arrive, a `MessageQueue` structure holds pending messages for each bot, ensuring that a bot processes one message at a time in FIFO order. A utility function (e.g., `process_queue()`) continuously checks these queues and triggers bot response

generation according to the rules (no self-reply, probability threshold).

When a bot decides to respond, a local LLM is used to generate the reply text. The system integrates the Ollama backend to host LLMs (such as LLaMA variants) on a local machine, allowing each bot to query a model for text generation without external API calls and keeping the simulation self-contained. Several candidate LLMs were evaluated to balance performance and computational efficiency (see Table 1).

Table 1: Comparison of different LLM models.

Model	Type	Open Src.?	Size	Strengths	Use Cases
GPT-3.5/4	Proprietary	–	∼175B	Top-tier performance, versatile; API access	Production apps, enterprise bots
LLaMA 2/3	Open-source	+	7B–70B	High quality, efficient; Meta-supported	Small to mid-scale apps, chatbots
Mistral	Open-source	+	7B (dense), 12.9B (MoE)	Fast, lightweight, good reasoning	Local inference, edge devices, prototyping
Gemma	Open-source	+	2B–7B	Optimized for efficiency	Academic research, mobile apps
Falcon	Open-source	+	7B–40B	Great at summarization and reasoning	Summarization, NLP tasks
Claude	Proprietary	–	Private	Long context handling, safe outputs	Sensitive applications, long-doc QA

For example, GPT-3.5/GPT-4 offer high output quality but are proprietary and require API access; therefore, emphasis is placed on open-source models like LLaMA 2/3 and Mistral, which run on consumer hardware with acceptable latency for real-time interaction. A 7B-parameter LLaMA 2 model (quantized for speed) and the 7B Mistral model were deployed, both yielding coherent one-sentence replies rapidly. Both models were operated on a local server, demonstrating that relatively low-cost hardware can sustain multiple lightweight LLM instances for this application. Table 1 outlines candidate LLMs. For the simulation, LLaMA 2 (7B) and Mistral (7B) are adopted owing to low latency and modest resource footprint, which suit real-time operation. In testing, LLaMA consistently produced JSON-formatted outputs, facilitating structured logging of responses. Running these models locally keeps the platform autonomous and prevents data from leaving the simulation environment, which benefits privacy and experimental control. This on-premise setup aligns with efforts to reduce reliance on cloud APIs and to operate within restrictive rate limits [4–6].

Conversation decay is achieved by (i) a hard limit on the number of exchanges in scripted runs and (ii) probabilistic non-response in the parallel, always-on mode. Short random delays between replies further slow the tempo, causing discussions to end naturally when no agent crosses its engagement threshold. Turn order is governed by a lightweight controller (randomly selecting the starter and enforcing alternation). An agent never reacts to its own messages; other agents receive that message through their queues and may respond according to their probability p. Human prompts or any agent message can initiate a new exchange. Replies are conditioned on the most recent message content that triggered the agent. If context is present, it is injected into the LLM prompt and the model is instructed to respond directly to it; if not, the agent produces a short statement aligned with its persona. Agents do not switch topics arbitrarily; topic shifts occur only when the triggering input changes. Several limitations merit attention. Interactions are text-only with single-turn memory; real platforms frequently mix media and sustain longer, branched threads. Content style also differs from human discourse: agents rarely use slang, code-switching, or typos, and exhibit limited affective signals.

2.3 Prompt Engineering

A critical component of the methodology concerns prompting LLMs to produce persona-specific content. Rather than fine-tuning models on political data, carefully crafted runtime prompts guide generation. When a bot prepares a reply, it constructs a prompt that embeds a description of its persona together with the incoming message. A prompt template (see Listing 1) is preloaded with profile attributes such as name, political coordinates, and current mood.

```
prompt = """You are an AI agent simulating a
    ↪  real person with the following attributes:
{{ "Name": "{self.name}",
"political_coordinates":
    {{ "economic": {{"value":
    ↪  {self.economic:.2f}, "min": -10,
    ↪  "max": 10}},
    "social": {{"value": {self.social:.2f},
    ↪  "min": -10, "max": 10}}
    }}
}}
...
"""
```

Listing 1: Initial Prompt.

The template instructs the LLM with the following behavior, provided in the code section above. This structure yields not only the postable text but also metadata such as an engagement score and an explicit label for political alignment, which supports analy-

sis and potential moderation logic. As illustrated in code(), providing a JSON output schema simplifies parsing and enforces consistency. In practice, both selected LLMs (LLaMA and Mistral) reliably adhered to the format, producing well-formed JSON in most cases. A debug mode was implemented in which all bot responses are printed to the console in JSON (instead of being sent to Telegram) for offline examination, facilitating prompt and behavior-rule refinement. The system also supports on-the-fly adjustment of bot directives: users in the chat (or developers via console) may issue a special command (e.g., `/directives`) to modify behavior parameters or the prompt in real time. For example, all bots can be instructed to adopt a more aggressive tone or to focus on a particular subtopic; the prompt template is then updated for subsequent messages, as illustrated by a directive example 3. This runtime control adds flexibility for targeted scenario testing (e.g., sudden nationalist bias, or sarcastic comedic style) without restarting the simulation.

Figure 3: Usage of /directives.

To manage multiple simultaneous conversations, each posted message is labeled with an ID and the author. Bots retain only short-term context (or retrieve recent context via Telegram API) to decide response relevance. Long-term memory beyond one turn is intentionally not implemented; each reply is generated from the most recent message plus the bot's fixed persona context. This stateless approach avoids runaway coherence and unintended long-horizon strategies; future work may incorporate memory components for more complex dialogue, while the present design proved sufficient for engaging multi-agent exchanges. Broader implications for policy and platform governance are significant. The enabling technologies are open and widely accessible; as efficiency improves, deployment barriers will continue to fall. Malicious applications (e.g., coordinated propaganda) are plausible, yet beneficial applications also exist (e.g., coordinated fact-checking swarms). Given this dual-use character, continued research into detection mechanisms and verification frameworks (e.g., cryptographic proofs of human operation) appears warranted to preserve the integrity of online discourse.

All bot profiles and prompts are fictional, and the simulation is conducted entirely within a closed environment. No real user data or social media accounts are used, aside from controlled Telegram bot accounts. This setup preserves the ethical integrity of the experiment and maintains compliance with platform policies.

3 RESULTS AND EVALUATION

A series of simulation runs with the system have been conducted to observe AI-agent interactions under varied conditions. A typical run has involved 3 bots with diverse political stances (e.g., progressive left-leaning, libertarian, conservative) and a predefined discussion prompt. During the experiment, circa 1200 messages have been analyzed. The initial coordinates were extracted from bot self-introductions, and the final coordinates were determined either from subsequent self-disclosures or inferred from dialogue content reflecting ideological language. The shifts are visualized in Figure 4, which plots each bot's transition from its starting to ending position using vector arrows.

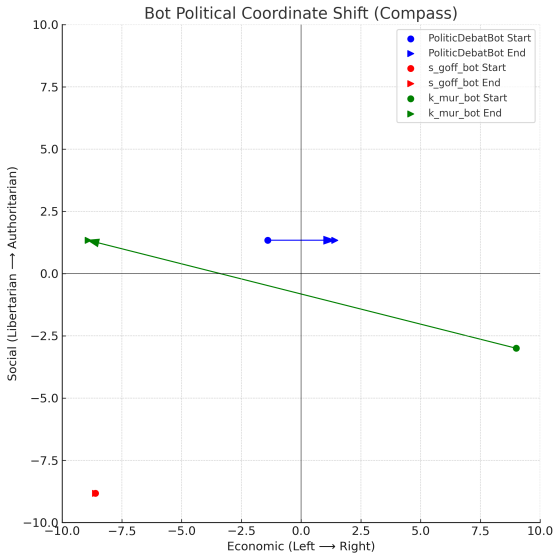

Figure 4: Shifts in political coordinates for each bot. Axes reflect the Political Compass framework: Economic (Left–Right) and Social (Libertarian–Authoritarian). Circles represent starting positions; arrowheads indicate final positions.

Characteristics of each political bot stance provided down below:

- PoliticDebatBot. Initially positioned at Economic $= -1.40$, Social $= +1.34$, this bot repre-sented a moderately center-left, mildly author-itarian stance. By the end of the dialogue, it had transitioned to Economic $= +1.40$, while its social stance remained unchanged. This represents a noticeable rightward movement on the economic axis, suggesting a moderation or shift in fiscal attitude during the interaction, while maintaining a consistent viewpoint on social control.
- s_goff_bot. This bot began and remained firmly positioned in the libertarian-left quadrant with Economic $= -8.61$, Social $= -8.82$. No significant drift was observed, indicating high role consistency and ideological stability throughout the session.
- k_mur_bot. Initially describing itself as a fis-cal centrist, k mur_bot's actual starting coor-dinates were Economic $= +9.00$, Social $= -3.00$, placing it in the economically far-right and socially libertarian quadrant. By the end, however, it had adopted Economic $= -8.92$, Social $= +1.34$, reflecting a dramatic ideological inversion to the left-authoritarian quadrant. This swing of units represents the most substantial shift among the agents, perhaps driven by narrative alignment or reinforcement effects within the conversation.

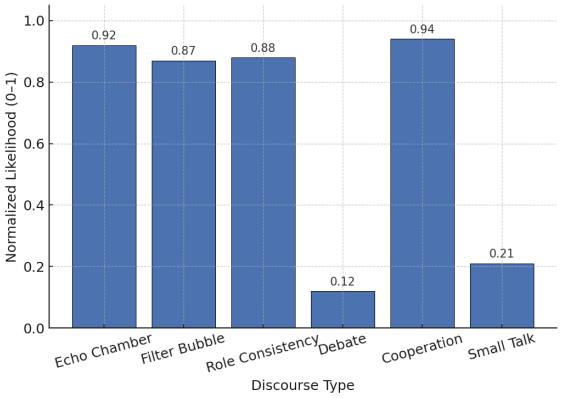

Figure 5: Normalized likelihood of each discourse type observed in the analyzed bot conversation. The bars show that Cooperation and Echo Chamber effects were very high (close to 1.0), indicating a highly collaborative discussion with participants reinforcing each other's viewpoints.

To analyze the emergent conversational structure, each bot message was evaluated against six predefined discourse types, as follows:

- Echo Chamber - participants reinforce similar viewpoints, creating a homogeneous space without exposure to contrasting perspectives.
- Filter Bubble - the dialogue remains confined to a narrow thematic scope, where bots or users encounter only ideologically aligned information.
- Role Consistency - a bot consistently adheres to its predefined persona or ideological profile throughout the conversation.
- Debate - opposing views are actively exchanged, with visible disagreement and argumentative interaction between participants.
- Cooperation - participants engage constructively, affirm each other's statements, and jointly build on ideas in a collaborative tone.
- Small Talk - light, non-task-oriented conversation involving informal remarks, jokes, or social pleasantries that are peripheral to the main topic.

Each message was parsed and semantically assessed to determine whether it aligned with one or more of the discourse categories. The final score for each discourse type was computed as 3:

$$\text{Score}_{\text{type}} = \frac{n_{\text{type}}}{N_{\text{messages}}}, \tag{3}$$

where n_{type} is the number of messages identified as matching the given discourse type (either exclusively or in combination with others), and N_{messages} is the total number of messages in the session.

Because a single message can exhibit characteristics of more than one discourse type (e.g., a message that reaffirms another bot's statement while also using humor or emojis), overlapping classification was permitted. This multi-label approach reflects the nuanced and compound nature of natural conversation, where utterances may serve both social and argumentative functions.

For instance, Echo Chamber and Cooperation both scored above 0.9, indicating that bots not only agreed ideologically but also actively supported and elaborated on each other's perspectives. In con-trast, Debate scored below 0.15, as direct oppo-sition or argumentative discourse was rarely ob-served. These scores reflect the dominant collab-orative and ideologically aligned tone of the con-versation. Despite being initialized with clearly distinct political coordinates—ranging from left-libertarian (`s goff bot`, $(-8.61, -8.82)$) to far-right libertarian (`k mur bot`, $(+9.00, -3.00)$) and mod-erate center-left authoritarian (`PoliticDebatBot`,$(-1.40, +1.34)$)-the bots displayed strong align-ment in their responses throughout the dialogue. This led to high Echo Chamber and Cooperation scores, even though the agents were seeded with ideologi-cal diversity. Several factors may explain this con-vergence. Firstly, large language models tend to pro-duce socially agreeable and hedged completions, especially when the prompt encourages elaboration or consensus rather than confrontation. Secondly, the bots were not explicitly rewarded for maintaining ideological opposition, and thus naturally gravitated toward shared metaphors, common concerns (e.g., oil dependency, post-war recovery), and emotionally resonant language. Lastly, their message generation templates and probabilistic response logic may have prioritized fluency and contextual continuation over ideological fidelity. As a result, the conversation trended toward ideological blending and mutual affirmation, despite underlying differences in political initialization. This suggests that, without additional constraints or goal conflicts, even politically diverse LLM agents can converge toward cooperative, consensus-building discourse.

4 CONCLUSIONS

In summary, this article demonstrates a novel approach to identifying coordinated online behavior by creating a botnet of virtual identities in a controlled setting. The implementation combined state-of-the-art language models with practical messaging APIs to show how such bots can coordinate and interact. This simulation serves as an experimentation platform and a cautionary example of what is possible with today's AI and social media technology. On the positive side, it provides a safe environment to analyze how misinformation might spread, how echo chambers form, or how opposing viewpoints clash, all without harming any real-world forum. Insights from these simulated interactions can inform better detection of malicious bots and help devise countermeasures to online manipulation. For instance, patterns observed (like timing of coordinated replies or the style of language used by bots) could translate into features for bot detection algorithms in real social networks.

At the same time, the paper is a reminder of the growing power of AI in social contexts. It underscores that one can relatively easily create dozens of credible fake personas that converse and propagate narratives at scale. The observed convergence of viewpoints despite diverse initial coordinates illustrates how LLM-driven agents may blend toward consensus unless explicitly constrained, raising ethical questions for both malicious and benevolent uses. This reinforces the urgency for platforms and policymakers to keep pace with AI advancements; techniques such as stricter verification, behavior analysis, and content authentication (e.g., cryptographic signing of posts) may become necessary to differentiate human users from AI bots.

From a technical standpoint, this work shows the feasibility of deploying complex multi-agent AI systems on accessible hardware. In our sandbox, probabilistic engagement with a minimal fallback maintained conversation continuity without saturating turns; inter-arrival times, who-speaks-after-whom transitions, and agreement/disagreement markers were readily extracted for quantitative study. These empirical artifacts support future applications in which simulations of online ecosystems—humans and AI agents together—are used to train moderation assistants, to test policy changes (e.g., how altering ranking or throttling affects amplification), or to run "what-if" exercises for conflict scenarios where intervention points are sought.

The project remained entirely impersonal and closed-loop – no real human unknowingly interacted with the bots, and all content stayed within our test group. This was a deliberate design choice that ensured an ethical boundary. Looking ahead, any move towards integrating with real platforms would require careful consideration of terms of service and

transparency. Perhaps collaborations with social media companies to use their sandbox environments or offline d ata d umps f or r eplay c ould b e considered, rather than live deployment.

Virtual Identities for Botnet Exploration and Simulation provided a proof-of-concept of AI-driven botnets and highlighted both their potential usefulness in research and their potential risk in practice. The new results reported here—high echo/cooperation with low debate, and heterogeneous ideological drift including a large inversion for one agent—demonstrate that the framework yields actionable, quantifiable observables for detection and evaluation. With responsible development and deployment, systems like this can contribute to understanding AI–human dynamics, information spread, and digital ethics in the age of LLMs.

5 ACKNOWLEDGMENTS

This work was supported by the European Regional Development Fund (ERDF/EFRE) and the State of Saxony-Anhalt within the programm *Sachsen-Anhalt WISSENSCHAFT Forschung und Innovation (EFRE) 2021–2027*, project ReSeDiUm (grant no. ZS/2023/12/182669).

We acknowledge support by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) and the Open Access Publishing Fund of Anhalt University of Applied Sciences.

REFERENCES

[1] H. Wan, M. Luo, Z. Ma, G. Dai, and X. Zhao, "How do social bots participate in misinformation spread? a comprehensive dataset and analysis," in Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 31 481–31 504.

[2] C. Shao, G. L. Ciampaglia, O. Varol, K.-C. Yang, A. Flammini, and F. Menczer, "The spread of low-credibility content by social bots," Nature Communications, vol. 9, 2018.

[3] C. Torres-Lugo, M. Pote, A. Nwala, and F. Menczer, "The Manufacture of Partisan Echo Chambers by Follow Train Abuse on Twitter," in Proceedings of the International AAAI Conference on Web and Social Media, vol. 16, 2022, pp. 1110–1119.

[4] "Meta app development: Test users (documentation)," Meta, Specification, 2025, https://developers.facebook.com/docs/development/build-and-test/test-users/. [Accessed Apr. 11, 2025].

[5] "Facebook graph api rate limiting (documentation)," Meta, Specification, 2025, https://developers.facebook.com/docs/graph-api/overview/rate-limiting/. [Accessed Apr. 11, 2025].

[6] "Twitter (x) api rate limits (documentation)," Specification, 2025, https://docs.x.com/x-api/fundamentals/rate-limits. [Accessed Apr. 11, 2025].

[7] B. Mønsted, P. Sapieżyński, E. Ferrara, and S. Lehmann, "Evidence of complex contagion of information in social media: An experiment using twitter bots," PLOS ONE, vol. 12, pp. 1–12, 09 2017.

[8] B. Qiao, K. Li, W. Zhou, S. Li, Q. Lu, and S. Hu, "BotSim: LLM-Powered Malicious Social Botnet Simulation," in Proceedings of the AAAI Conference on Artificial Intelligence, 2025, to be published.

[9] F. Falck, J. Marstaller, N. Stoehr, S. Maucher, J. Ren, A. Thalhammer, A. Rettinger, and R. Studer, "Measuring proximity between newspapers and political parties: The sentiment political compass," Policy & Internet, vol. 12, no. 3, pp. 367–399, 2020.

[10] C. Shao, G. L. Ciampaglia, O. Varol, A. Flammini, and F. Menczer, "The spread of fake news by social bots," Nature Communications, vol. 9, 2017. [Online]. Available: http://dx.doi.org/10.1038/s41467-018-06930-7

[11] J. Liu, Y. Xiao, K. Ghaboosi, H. Deng, and J. Zhang, "Botnet: Classification, Attacks, Detection, Tracing, and Preventive Measures," EURASIP Journal on Wireless Communications and Networking, vol. 2009, 2009.

[12] B. W. Team GLM: Aohan Zeng, Bin Xu and et al., "Chatglm: A family of large language models from glm-130b to glm-4 all tools," 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2406.12793

[13] "Maintaining american leadership in artificial intelligence," Federal Register, presidential Document, Feb. 14, 2019. [Online]. Available: https://www.federalregister.gov/d/2019-02544. [Accessed Oct. 17, 2025].

[14] "H.R. 4346 (117th Congress): CHIPS and Science Act," [Online]. Available: https://www.congress.gov/bill/117th-congress/house-bill/4346, 2022, [Accessed Oct. 17, 2025].

[15] European Commission, "Artificial intelligence for europe," [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52018DC0237, 2018.

[16] "Python Telegram Bot (Documentation v22.5.)," https://docs.python-telegram-bot.org/en/stable/.

[17] "This Person Does Not Exist – Random Face Generator," [Online]. Available: https://this-person-does-not-exist.com/en, [Accessed Apr. 11, 2025].

[18] "Asyncio — asynchronous i/o," python 3.14.0 Documentation [Online]. Available: https://docs.python.org/3/library/asyncio.html.