EVOLVING LLMS THROUGH TEXT-BASED SELF-PLAY: ACHIEVING EMERGENT PERFORMANCE

Eric Martin
Independent Researcher
ermartin86@gmail.com

ABSTRACT

We introduce a scalable, text-only self-play framework that evolves large language models (LLMs) on a single 6GB GPU, using TinyLlama without fine-tuning or human feedback. This lightweight alternative to resource-intensive RL achieves an 89.4% win rate ($p < 0.001$) against the baseline in 500 games after 67 iterations in 47 hours, offering a flexible testbed for emergent intelligence in multi-agent language scenarios.

1 Introduction

While AlphaZero mastered Go with millions of simulations, real-world scenarios require LLMs to navigate linguistic nuance, not just rigid rules—yet scalable methods to evolve such skills in large language models (LLMs) remain elusive. Self-play has revolutionized AI, enabling breakthroughs like AlphaGo's mastery of Go [1] and AlphaZero's dominance in Chess and Shogi [2]. These systems thrive in structured environments with fixed rules, yet they struggle to capture the fluid, text-based strategies of human discourse, often requiring 10^6–10^8 FLOPs for reinforcement learning (RL)-based approaches, limiting accessibility. In contrast, our method runs on a single 6GB GPU, democratizing access to LLM evolution. Real-world scenarios—e.g., diplomatic negotiations or online forums—thrive on linguistic ambiguity, a domain where LLMs can excel if guided by competitive evolution. Here, emergent intelligence refers to novel strategies arising from iterative self-play without explicit design.

We propose a framework where LLMs evolve via self-play in a text-based game, blending competition and cooperation to foster emergent strategies. This language-driven approach offers a lightweight alternative to RL, relying solely on the base LLM's pretraining without supervised fine-tuning or human feedback. It captures the strategic ambiguity of human communication, an underexplored frontier in game AI.

In our proof-of-concept (POC), bots compete on a 7×7 grid, refining strategies over iterations. Using TinyLlama, we conducted 67 iterations in 47 hours, achieving an 89.4% win rate against the initial TinyLlama baseline in 500 randomized games. This text-only system lays a foundation for studying emergent intelligence in language-driven simulations. Our contributions are: (1) a text-only multi-agent self-play system; (2) demonstrated performance gains in a competitive text game; and (3) a scalable testbed for emergent intelligence.

2 Related Work

LLM applications in games set the stage for our multi-agent self-play approach. Tsai et al. [3] focus on narrative generation in solo text adventures. In contrast, our framework uses competitive self-play to drive strategy emergence. Cheng et al. [4] apply self-play to enhance reasoning in a single-agent setting, while Gao et al. [5] explore multi-agent coordination without competitive improvement. Unlike RL-driven systems [6, 8], which rely on hand-crafted rewards and high compute, we use a perturbation-based approach—random weight tweaks—to evolve strategies. Our perturbation mimics evolutionary strategies, trading RL's complexity for simplicity and scalability. Inspired by AlphaZero's self-play [2], our text-driven method integrates competition and cooperation with minimal resources, leveraging TinyLlama [7] as a lightweight tool.

3 Methodology

3.1 Game Design

Bots compete on a 7×7 grid to harvest resources and clone, winning individually (3 clones) or collectively (20+ clones, $\sim$50% grid occupancy). The game unfolds on a 7×7 grid (9×9 with boundary wall 'u', an impassable barrier), where bots ('x' for self, 'z' for rivals, 't' for talking bots that can communicate) harvest wheat ('w', 10 seeds), push rocks ('r'), plant seeds ('v', matures to wheat in 10 turns), and move. Bots aim to clone 3 times individually (individual win) or produce 20+ clones collectively, a threshold chosen because it marks significant game progress (over half the spaces occupied), with the team (control vs. variable) producing more clones declared victorious. Starting with 100 energy and 0 seeds, bots lose 1 energy per move (except cloning, costing 75 energy and requiring an adjacent bot). Energy depletion causes death. Actions are listed in Table 1, with text messages enabling communication. A 7×7 grid was chosen as small enough for TinyLlama's context window yet complex enough for strategy. Figure 1 shows a sample mid-game state (global view, not bot-visible).

Action	Energy Cost	Prerequisites
Move (Up/Down/Left/Right)	1	None
Eat (10 seeds $\rightarrow$ 10 energy)	1	Adjacent wheat or seeds held
Plant (1 seed $\rightarrow$ 1 wheat)	1	Seeds held
Push (rock)	1	Adjacent rock
Mate (clone)	75	Adjacent bot, 75+ energy

Table 1: Bot actions, energy costs, and prerequisites

```
u u u u u u u u u
u 0 0 0 w r 0 z u
u 0 0 0 0 v 0 0 u
u 0 0 0 0 v 0 z u
u r t 0 0 v t z u
u 0 0 0 w z z t u
u 0 t t t t z r u
u 0 0 r z 0 z r u
u u u u u u u u u
```

Figure 1: Sample 7×7 grid mid-game (global view), showing bots (z: rivals, t: talking), resources (w: wheat, v: seeds, r: rocks), walls (u), and empty spaces (0). Bots see only local states.

3.2 LLM Control

Bots use TinyLlama-1.1B-Chat-v1.0 (8-bit quantized) [7] to process grid states as text, outputting moves ("Up", "Down", "Left", "Right"), actions ("Eat", "Plant", "Mate"), or messages ("Talk: [text]"). Invalid commands trigger a two-step response: first, a check for simplified command variants (e.g., extracting key letters like "Mo" for "Move" or "Ea" for "Eat" from noisy output), then, if unresolved, a heuristic fallback prioritizes mating if energy exceeds 75 and a bot is adjacent, otherwise eating if wheat or seeds are available, moving toward wheat if visible, planting if seeds are held, or moving randomly.

3.3 Self-Play Mechanism

Each iteration pits a control LLM against a perturbed clone in a multi-agent setup: two control bots vs. two variable bots. Weights are perturbed every iteration using a bimodal distribution—mixing small (0–1%, $\mathcal{N}(0, 0.01)$) and large (1–20%, $\mathcal{N}(0, 0.2)$) changes—where small changes refine local optima and large changes escape plateaus, balancing refinement and exploration. The variable replaces the control if it wins $\geq 4/5$ games, ensuring consistent improvement. This mirrors evolutionary pressures and supports scalability unlike resource-heavy RL.

4 Experiments

4.1 Setup

Experiments used TinyLlama-1.1B-Chat-v1.0 (8-bit) on a 6GB NVIDIA RTX GPU, 64GB RAM, AMD Ryzen 7 7700 (3.80 GHz) system running Windows, with Python 3.12+ and PyTorch 2.6+ in a Conda environment. Randomized 7×7 maps featured 20 wheat tiles, 5 rocks, and 4 initial bots (2 control, 2 variable).

4.2 Results

Over 47 hours, 67 iterations emerged, each outperforming its predecessor in $\geq 4/5$ games. In a 500-game showdown with starting positions swapped for half to control for positional bias, the 67th iteration won 447 games (89.4%) vs. 44 (8.8%) for the initial TinyLlama baseline, with 9 draws (1.8%) (Table 2). A t-test (`scipy.stats.ttest_ind`, `equal_var=False`, `alternative='less'`) yielded $t = -43.0$, $p < 0.001$, confirming significant improvement.

Model	Games	Wins	Losses	Draws
Initial TinyLlama	500	44 (8.8%)	447	9
67th Iteration	500	447 (89.4%)	44	9

Table 2: 500-game showdown results ($p < 0.001$).

4.3 Emergent Behaviors

TinyLlama's limited capacity limited complex interactions, but the 67th iteration showed behavioral shifts. Sample outputs include (non-English characters excluded for ease of formatting):

- Bot 0 at (4,2), energy 100, seeds 0: 'Down' from 'mars previous marsbar marsirabarbar SSwend'
- Bot 9 at (5,6), energy 2, seeds 6: 'Plant' from 'SSbarvoy mars Preisbarjkbar'
- Bot 20 at (6,6), energy 99, seeds 57: 'Mate' from 'mvbar indic SSbar mars anni marserebar'

Noisy outputs likely stem from lengthy instruction prompts, TinyLlama's 2048-token limit, or pretraining biases, embedding valid commands within incoherent text rather than strategic messaging.

5 Discussion and Future Work

This framework provides a scalable sandbox for probing emergent intelligence in LLMs. Scaling to a 50×50 grid (e.g., simulating 100 bots in resource-scarce conditions) with a 13B-parameter model could unlock richer strategies. TinyLlama's 1.1B parameters limit reasoning depth, as seen in its lack of coherent messaging. Larger models might enable trust or deception dynamics.

5.1 Ethical Considerations

Emergent deception, while a research goal, requires monitoring to ensure trustworthiness. For example, bots might misreport wheat at (3,3) to starve rivals, a tactic absent in TinyLlama but plausible with larger models. This ties to TinyLlama's incoherent messaging, which lacks such intent but hints at future risks. Bots could also unfairly sway team decisions in games or inflate perceived resource scarcity in simulations, impacting trust in AI systems. PPO-based reward shaping [8] and oversight could help align behaviors with human values.

5.2 Future Directions

- **Scale:** Test a 50×50 grid for resource-scarce multi-agent dynamics, hypothesizing that increased competition drives novel strategies.
- **Models:** Employ advanced LLMs (e.g., LLaMA-70B), hypothesizing that a 13B model might enable alliance formation due to a $\sim$10x increase in reasoning capacity.
- **RL Integration:** Swap perturbation for PPO or Q-learning [9].
- **Message Coherence:** Assess communication more fully as a strategy.

6 Conclusion

We present a self-evolving LLM framework, validated by 67 TinyLlama iterations with an 89.4% win rate. This scalable system pushes forward multi-agent learning, language evolution, and autonomous decision-making in text-driven settings. Code: `https://github.com/emartin59/text-game-llm-improver`.

Acknowledgments

Thanks to the open-source community for TinyLlama [7] and DeepMind's AlphaZero [2] for inspiration.

References

[1] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, vol. 529, pp. 484–489, 2016. `https://doi.org/10.1038/nature16961`.

[2] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis, "A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-Play," *Science*, vol. 362, pp. 1140–1144, 2018. `https://doi.org/10.1126/science.aar6404`.

[3] Chen Feng Tsai, Xiaochen Zhou, Sierra S. Liu, Jing Li, Mo Yu, and Hongyuan Mei, "Can Large Language Models Play Text Games Well?" *arXiv:2304.02868*, 2023. `https://arxiv.org/abs/2304.02868`.

[4] Pengyu Cheng, Tianhao Hu, Han Xu, Zhisong Zhang, Zheng Yuan, Yong Dai, Lei Han, Nan Du, and Xiaolong Li, "Self-playing Adversarial Language Game Enhances LLM Reasoning," *arXiv:2404.10642*, 2024. `https://arxiv.org/abs/2404.10642`.

[5] Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li, "Large Language Models Empowered Agent-based Modeling and Simulation," *arXiv:2312.11970*, 2023. `https://arxiv.org/abs/2312.11970`.

[6] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d.O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang, "Dota 2 with Large Scale Deep Reinforcement Learning," *arXiv:1912.06680*, 2019. `https://arxiv.org/abs/1912.06680`.

[7] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu, "TinyLlama: An Open-Source Small Language Model," *arXiv:2401.02385*, 2024. `https://arxiv.org/abs/2401.02385`.

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov, "Proximal Policy Optimization Algorithms," *arXiv:1707.06347*, 2017. `https://arxiv.org/abs/1707.06347`.

[9] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller, "Playing Atari with Deep Reinforcement Learning," *arXiv:1312.5602*, 2013. `https://arxiv.org/abs/1312.5602`.