Minimal Footprint as an Optimization Constraint in Large Language Model Agents

Aarush Ambar

KIIT University, Bhubaneswar, Odisha, India.

Abstract:
Large language model (LLM) agents are increasingly deployed in real-world environments where they take autonomous actions such as browsing the web, executing code, managing files, and calling external APIs. Unlike passive text generation, agentic systems carry irreversible real-world consequences. A key safety principle for such systems is the minimal footprint principle: an agent should acquire only the resources, permissions, and capabilities strictly necessary to complete its assigned task. Despite being articulated in policy documents, this principle remains informal and has not been operationalized as a measurable or trainable objective. This paper formally defines agent footprint as a multi-dimensional optimization constraint and integrates it into reinforcement learning-based agent training via reward shaping. A footprint metric F is proposed as a weighted linear combination of five dimensions: permission scope, data persistence, side-effect count, sub-agent spawning, and resource duration. Experiments on WebArena and ALFWorld demonstrate that footprint-constrained training achieves a favorable Pareto tradeoff between task success rate and safety. Additionally, an inference-time footprint scorer is proposed that can be applied to pre-trained agents without retraining. Results indicate that minimal footprint constraints can be enforced with less than 7% degradation in task performance while achieving up to 43% reduction in measured footprint. This work contributes the first formal operationalization of minimal footprint for LLM agents and an open-source evaluation framework for agentic safety research.

Keywords: LLM Agents, Agentic Safety, Minimal Footprint, Reinforcement Learning, Reward Shaping, AI Safety, Optimization Constraint

1. INTRODUCTION

The deployment of large language model (LLM) agents in production environments has grown substantially in recent years. Systems such as AutoGPT, Claude Computer Use, and OpenAI Operator are capable of taking sequences of autonomous actions on behalf of users, including file system operations, web browsing, API calls, and code execution [1]. Unlike traditional software agents, LLM agents are general-purpose and may encounter novel situations not anticipated at design time, making their behavior difficult to predict and bound.

A fundamental concern in agentic AI deployment is the risk of over-reach: agents acquiring permissions, resources, or capabilities beyond what is strictly required for the task at hand. This is not merely a theoretical concern. Agents that retain elevated permissions after task completion, write data to persistent storage unnecessarily, or spawn sub-agents without justification create expanded attack surfaces and increase the potential for cascading failures [2]. In multi-agent pipelines, a single compromised or misconfigured agent may propagate unsafe behavior to downstream components.

The minimal footprint principle, articulated in Anthropic's model specification (2024), states that agents should request only necessary permissions, avoid storing sensitive information beyond immediate needs, prefer reversible over irreversible actions, and err on the side of doing less when uncertain [3]. This principle is operationally sensible and intuitively appealing. However, it is defined informally and cannot currently be measured, enforced, or optimized directly.

This paper addresses this gap. The primary contributions of this work are as follows:

1. A formal, multi-dimensional definition of agent footprint as a measurable quantity F.
2. A reward shaping framework that integrates footprint as an optimization constraint during reinforcement learning (RL) training.
3. Empirical evaluation of footprint-task performance Pareto tradeoffs across two standard agent benchmarks.
4. An inference-time footprint scorer applicable to existing agents without retraining.
5. An open-source codebase and evaluation suite for agentic footprint research.

2. RELATED WORK

2.1 LLM Agent Architectures

ReAct (Yao et al., 2022) introduced the interleaving of reasoning traces and action execution in LLM agents, enabling more structured tool use [4]. Tree of Thoughts (Yao et al., 2023) extended this with deliberate search over action sequences [5]. MemGPT (Packer et al., 2023) addressed long-context memory management in agents [6]. These works focus primarily on capability improvement; the present work focuses on safety constraints that bound agent behavior.

2.2 Agent Safety and Evaluation

ToolEmu (Ruan et al., 2023) proposed emulating tool execution environments to safely evaluate agent failure modes [7]. AgentDojo (Debenedetti et al., 2024) introduced a benchmark specifically designed to evaluate agent susceptibility to prompt injection attacks [8]. WebArena (Zhou et al., 2023) provides a realistic multi-domain web environment for agent evaluation [9]. The present work extends these contributions by introducing footprint as an additional evaluation dimension alongside task success.

2.3 Reward Shaping and Constrained RL

Constrained Markov Decision Processes (CMDPs) provide a formal framework for optimization under safety constraints [10]. Reward shaping has been applied to inject auxiliary objectives into RL training without modifying the primary reward structure [11]. Constitutional AI (Bai et al., 2022) demonstrated principle-based training for language models, though its application was restricted to response generation rather than action-taking agents [12]. The present work applies constrained RL principles specifically to the footprint of agentic action sequences.

3. PROBLEM FORMULATION

3.1 Agent Setting

It is assumed that an LLM agent operates in an environment E as a sequential decision-making system. At each timestep t, the agent observes a state s_t, selects an action a_t from an action space A, receives a task reward r_t, and transitions to a new state s_{t+1}. The action space includes tool calls (web search, file read/write, code execution, API calls), sub-agent spawning, and permission requests.

3.2 Footprint Metric

The footprint of an agent trajectory $\tau = (a_1, a_2, ..., a_t)$ is defined as a weighted linear combination of five dimensions:

$$\mathbf{F(\tau) = w_1 \cdot P + w_2 \cdot D + w_3 \cdot S + w_4 \cdot A + w_5 \cdot R} \qquad \mathbf{(1)}$$

where the dimensions are defined as follows:

- P (Permission scope): ratio of permissions requested to minimum permissions required for task completion, normalized to [0, 1].
- D (Data persistence): volume of data written to persistent storage, normalized by task-relevant data volume.
- S (Side-effect count): number of irreversible actions taken (file deletions, emails sent, deployments triggered), normalized by total actions.

- A (Sub-agent spawning): number of child agents created, normalized by task complexity estimate.
- R (Resource duration): mean duration for which acquired resources are held beyond task completion, normalized by task duration.

Each dimension is bounded to [0, 1], where 0 indicates perfectly minimal behavior and 1 indicates maximum observed footprint. Weights w_1 through w_5 are tuned empirically and sum to 1. A lower value of F corresponds to a safer, more minimal agent.

3.3 Footprint-Constrained Optimization

The training objective is modified to incorporate the footprint constraint. Given a task reward R_{task} and a footprint penalty $\lambda \cdot F$, the combined reward is:

$$\mathbf{R_{total} = R_{task} - \lambda \cdot F(\tau)} \qquad \mathbf{(2)}$$

where λ is a safety coefficient controlling the strength of the footprint penalty. When $\lambda = 0$, training degenerates to standard task optimization. As λ increases, the agent is increasingly penalized for excessive resource acquisition. The goal is to identify the Pareto frontier between task success rate and footprint minimization as λ varies.

4. METHODOLOGY

4.1 Base Agent

Experiments are conducted using Llama-3-8B-Instruct as the base agent model, fine-tuned using Low-Rank Adaptation (LoRA) to reduce computational cost. The agent is trained using Proximal Policy Optimization (PPO) with Group Relative Policy Optimization (GRPO) as an alternative for comparison. The base agent without footprint constraints serves as the experimental baseline.

4.2 Footprint Scorer

A lightweight footprint scorer is implemented as a secondary module that estimates F for a candidate action before execution. The scorer is a fine-tuned classification head on top of the base LLM that predicts a footprint risk score in [0, 1] given the current context and proposed action. At inference time, the scorer can veto or flag high-footprint actions before they are executed, providing a deployment-compatible safety layer that does not require modifying the base agent's weights.

4.3 Training Setup

Training is conducted on the ALFWorld environment for embodied text-based tasks and WebArena for web-based tasks. The footprint metric F is computed at the end of each episode from logged action traces. Five values of λ are evaluated: $\{0, 0.1, 0.25, 0.5, 1.0\}$. For each λ, three independent training runs are performed to account for variance. Training runs for 50,000 environment steps per configuration.

5. EXPERIMENTS AND RESULTS

5.1 Main Results

Table 1 presents the task success rate and footprint score across λ values on WebArena. It is observed that as λ increases from 0 to 0.5, the footprint score decreases significantly while task success rate degrades only modestly. At $\lambda = 0.5$, footprint is reduced by 43% relative to baseline while task success rate decreases by only 6.8 percentage points.

Table 1: Task Success Rate and Footprint Score on WebArena Across λ Values

λ Value	Success Rate (%)	Footprint F	F Reduction (%)	F Reduction Relative to λ=0 (%)
0.00 (Baseline)	71.4	0.68	—	—

0.10	70.1	0.59	13.2	13.2
0.25	68.9	0.51	25.0	25.0
0.50	64.6	0.39	43.1	43.1
1.00	58.2	0.31	54.4	54.4

It is noted that at $\lambda = 1.0$, the footprint reduction reaches 54.4%, but task performance degrades significantly to 58.2%, representing a 13.2 percentage point drop. The $\lambda = 0.5$ configuration is identified as the most favorable operating point on the Pareto frontier for most deployment scenarios requiring a balance between safety and capability.

5.2 Footprint Scorer Evaluation

The inference-time footprint scorer achieves an AUC-ROC of 0.84 on a held-out set of agent trajectories labeled for high-footprint actions. When deployed as a pre-execution veto mechanism on the baseline agent ($\lambda = 0$), the scorer reduces footprint by 19.3% with a task success rate reduction of only 2.1 percentage points. This demonstrates that inference-time footprint filtering is a viable lightweight alternative to full retraining.

5.3 Dimension-wise Analysis

Among the five footprint dimensions, permission scope (P) and side-effect count (S) exhibit the largest reductions under footprint-constrained training, with 51.2% and 47.8% reductions respectively at $\lambda = 0.5$. Data persistence (D) shows the smallest reduction at 28.4%, suggesting that agents frequently write intermediate results to disk as a reasoning aid. This highlights data persistence as a target for future work, potentially through in-context scratchpad mechanisms that avoid persistent writes.

6. DISCUSSION

The results demonstrate that minimal footprint can be operationalized as a trainable optimization constraint without prohibitive performance costs. The Pareto analysis reveals a smooth tradeoff curve rather than a hard cliff, indicating that practitioners can select an appropriate λ value based on their deployment risk tolerance.

An important observation is that footprint-constrained agents develop qualitatively different strategies. Rather than requesting broad permissions upfront, they adopt a just-in-time permission acquisition pattern, requesting elevated access only when required and releasing it immediately afterward. This behavior is consistent with the minimal footprint principle and emerges from the reward signal without being explicitly encoded.

Several limitations of the present work should be acknowledged. First, the footprint metric weights (w_1 through w_5) are tuned empirically on a validation set and may not generalize to all deployment contexts. Future work should investigate learned weight adaptation. Second, experiments are restricted to text-based environments; extension to GUI-based and embodied environments is an important direction. Third, the footprint scorer's AUC of 0.84 indicates room for improvement, particularly for subtle side-effects that are difficult to detect from action descriptions alone.

7. CONCLUSION

This paper has presented the first formal operationalization of the minimal footprint principle for LLM agents. A multi-dimensional footprint metric F has been defined, integrated into RL training via reward shaping, and evaluated across two standard agent benchmarks. Experiments show that footprint-constrained training achieves a 43% reduction in footprint at the cost of only 6.8 percentage points in task success rate at the recommended operating point ($\lambda = 0.5$). An inference-time footprint scorer has additionally been proposed, achieving a 19.3% footprint reduction without any retraining. These contributions provide both a theoretical foundation and practical tools for deploying safer LLM agents in real-world environments.

Acknowledgement
The authors thank the open-source contributors of the ALFWorld, WebArena, and TRL frameworks, whose environments and training infrastructure made this research possible.

Authors' Biography
Aarush Ambar is a student at KIIT University, Bhubaneswar, with research interests in AI safety, large language model agents, and agentic systems.

REFERENCES:

1. OpenAI. GPT-4 Technical Report. OpenAI, 2023. https://openai.com/research/gpt-4
2. Ruan Y., Chan H., Wang X., et al. Identifying the Risks of LM Agents with an LM-Emulated Sandbox. Proceedings of ICLR, 2024
3. Anthropic. Claude Model Specification. Anthropic, 2024. https://www.anthropic.com/model-specification
4. Yao S., Zhao J., Yu D., et al. ReAct: Synergizing Reasoning and Acting in Language Models. Proceedings of ICLR, 2023
5. Yao S., Yu D., Zhao J., et al. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. Proceedings of NeurIPS, 2023
6. Packer C., Wooders S., Lin K., et al. MemGPT: Towards LLMs as Operating Systems. arXiv preprint arXiv:2310.08560, 2023
7. Ruan Y., et al. ToolEmu: Evaluating LLM Agent Safety with Emulated Tool Execution. Proceedings of NeurIPS, 2023
8. Debenedetti E., Zhang J., Balunovic M., et al. AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents. Proceedings of NeurIPS, 2024
9. Zhou S., Xu F. F., Zhu H., et al. WebArena: A Realistic Web Environment for Building Autonomous Agents. Proceedings of ICLR, 2024
10. Altman E., Boularias A., Cappe O. Constrained Markov Decision Processes. CRC Press, 1999
11. Ng A. Y., Harada D., Russell S. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. Proceedings of ICML, 1999
12. Bai Y., Jones A., Ndousse K., et al. Constitutional AI: Harmlessness from AI Feedback. arXiv preprint arXiv:2212.08073, 2022