File 2510.04618, filed 6 October 2025, section: Context & memory

Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu et al.. Stanford, SambaNova, UC Berkeley

In brief

ACE treats an agent's context as an evolving playbook that is generated, reflected on and curated over time, improving agents without weight updates.

Why builders should care

The core context-engineering paper: structured, incremental context updates instead of rewriting prompts wholesale.

The paper, in plain English

Many LLM applications improve by adapting their context (instructions, strategies, evidence) rather than their weights. Existing approaches tend to suffer from brevity bias, dropping domain insight for short summaries, and context collapse, where repeated rewriting erodes detail.

ACE treats context as an evolving playbook that accumulates, refines and organizes strategies through generation, reflection and curation, using structured incremental updates instead of full rewrites.

It optimizes both offline context (system prompts) and online context (agent memory). The abstract reports +10.6% on agent benchmarks and +8.6% on finance over strong baselines, with lower adaptation latency and rollout cost, and it can learn from natural execution feedback without labels. On the AppWorld leaderboard it matches the top-ranked production agent overall and beats it on the harder split, using a smaller open-source model.

How to apply it

1. Replace 'rewrite the prompt' with 'append and refine a playbook' of strategies.
2. Separate the roles: one step generates, one reflects on outcomes, one curates what stays.
3. Feed in execution signals you already have, like task success and errors, as the learning signal.
4. Watch for context collapse: diff your playbook over time and make sure detail is not being lost.

Key results

Quoted from the abstract

> *Across agent and domain-specific benchmarks, ACE optimizes contexts both offline (e.g., system prompts) and online (e.g., agent memory), consistently outperforming strong baselines: +10.6% on agents and +8.6% on finance, while significantly reducing adaptation latency and rollout cost.*

> *Notably, ACE could adapt effectively without labeled supervision and instead by leveraging natural execution feedback.*

> *On the AppWorld leaderboard, ACE matches the top-ranked production-level agent on the overall average and surpasses it on the harder test-challenge split, despite using a smaller open-source model.*

Summary written from the arXiv abstract. Read the original: https://arxiv.org/abs/2510.04618

Curated by Atharv Pawar