Long-Context Management in LLM Agents

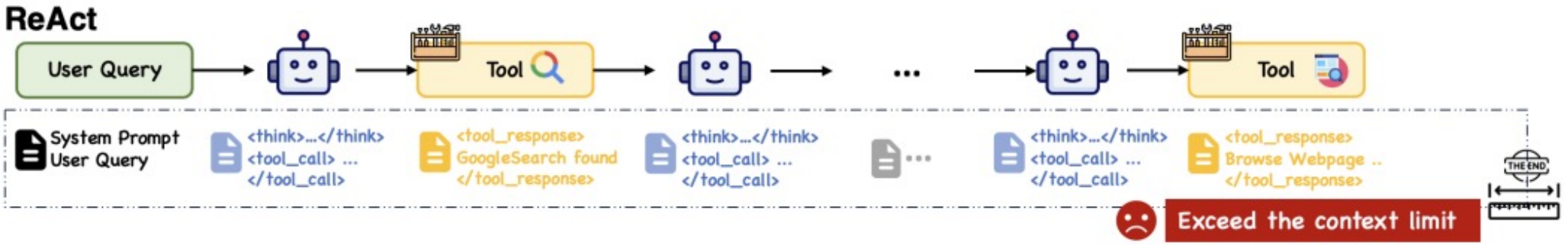
2026 하계 세미나
정지민

Natural Language Processing
& Artificial Intelligence

Background

Why Context Management is Important in LLM Agents?

- Agent는 단순 LLM 호출이 아니라, 여러 step의 reasoning, tool call, observation, code 실행 결과, user instruction, memory 등을 계속 누적함

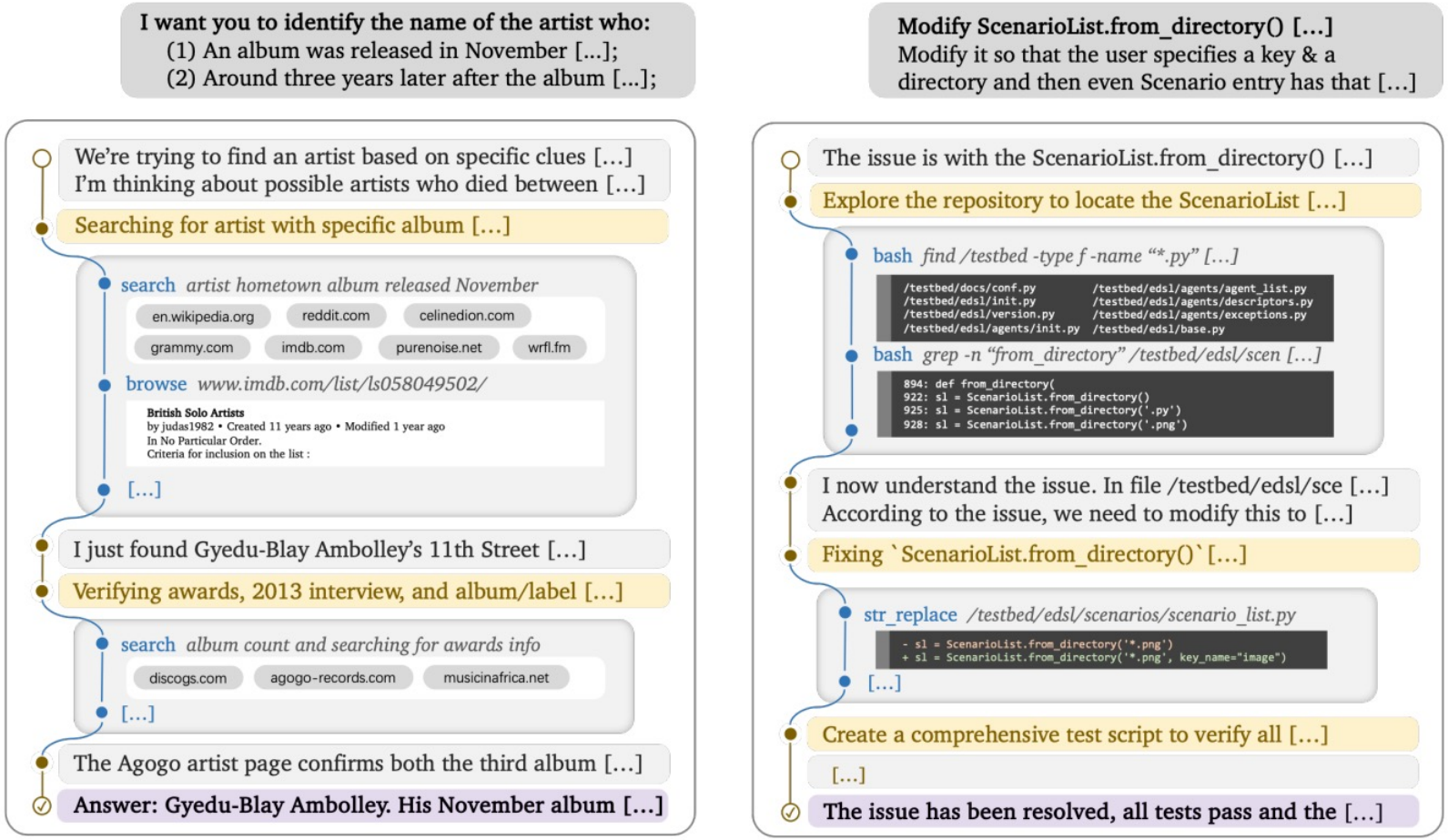


- 긴 context를 그냥 유지하면 비용 증가, retrieval/attention noise, 오래된 정보와 최신 정보의 충돌 문제가 생김
- Claude Code와 같은 agent도 긴 작업 중 context를 요약/정리하면서 이어가는 방식이 필요함

- 그렇다면 agent의 context는 단순히 “길게 넣는 것”이 아니라 어떻게 설계, 갱신, 압축해야 하는가?

Background

Context Management in Real Agents



AGENTIC CONTEXT ENGINEERING: EVOLVING CONTEXTS FOR SELF-IMPROVING LANGUAGE MODELS

Qizheng Zhang[1*], **Changran Hu**[2*], **Shubhangi Upasani**[2], **Boyuan Ma**[2], **Fenglu Hong**[2], **Vamsidhar Kamanuru**[2], **Jay Rainton**[2], **Chen Wu**[2], **Mengmeng Ji**[2], **Hanchen Li**[3], **Urmish Thakker**[2], **James Zou**[1], **Kunle Olukotun**[1]

[1]Stanford University [2]SambaNova Systems, Inc. [3]UC Berkeley

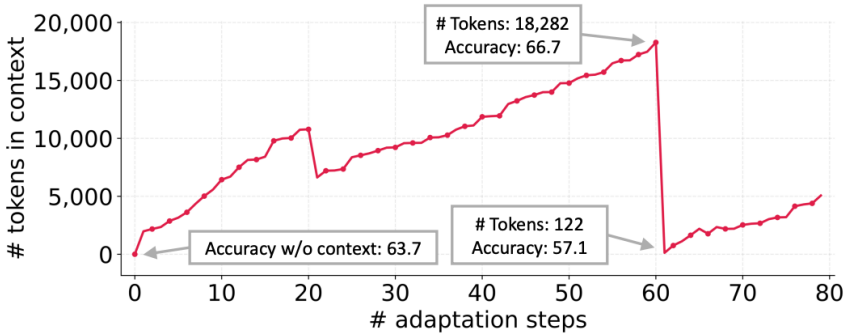
{qizhengz,kunle}@stanford.edu changran_hu@berkeley.edu

ace-agent/ace ace-agent.github.io [*]Equal contribution.

ICLR 2026

Context Adaptation as a New Paradigm

- 최근 LLM agent는 모델 가중치를 직접 수정하기보다, 입력 컨텍스트를 조정해 성능을 높임
- 한계 1 (Brevity Bias): 짧고 간결한 prompt를 선호하는 과정에서 중요한 세부 내용이 사라질 수 있음
- 한계 2 (Context Collapse): LLM에 의한 단일 재작성에 의존하는 방법에서 시간이 지남에 따라 더 짧은 요약으로 퇴화하는 현상이 발생할 수 있음



Agentic Context Engineering (ACE)

- Contest는 단순 요약이 아니라, 전략을 축적하고 정리하는 구조화된 playbook 이어야 함
- ACE는 generation, reflection, curation을 통해 context를 점진적으로 개선함
- 핵심 목표는 세부적인 도메인 지식을 보존하면서 self-improving agent를 가능하게 하는 것

Introduction

Contribution

- ACE는 strong baseline을 일관되게 능가하며, offline 및 online adaptation 설정 모두에서 에이전트에 대해 평균 10.6%, 도메인별 벤치마크에 대해 평균 8.6%의 성능 향상을 가져옴

- ACE는 lable이 지정된 supervision 없이도 효과적인 컨텍스트를 구성할 수 있으며, 대신 실행 피드백과 환경 신호를 활용함

- AppWorld 벤치마크 리더보드에서 오픈 소스 모델인 DeepSeek-V3.1을 사용하면서도 최고 순위의 production 수준 에이전트인 IBM-CUGA(GPT-4.1 기반)을 능가함

- ACE는 기존 adaptative 방법보다 훨씬 적은 roll-out과 낮은 latency를 달성하며, 확장 가능한 자체 개선이 더 높은 정확도와 더 낮은 비용으로 달성될 수 있음을 보여줌

Background and Motivation

Context Adaptation

- 최근 context adaptation 방법들은 실행 trace, 추론 단계, 검증 결과 등을 바탕으로 자연어 피드백을 생성하고, 이를 다시 context에 반영하여 반복적으로 성능을 개선함
- 대표적으로 **Reflextion**은 reflection을 통해 에이전트 계획을 개선하고, **TextGrad**와 **GEPA**는 텍스트 피드백이나 실행 trace를 활용해 프롬프트를 최적화함. **Dynamic Cheatsheet**는 과거 성공, 실패에서 전략과 교훈을 축적하는 외부 메모리를 구성함

Limitations of Existing Context Adaptation Methods

- Brevity Bias: 탐색 공간을 좁힐 뿐만 아니라, 최적화된 프롬프트가 종종 초기 프롬프트의 동일한 결함을 상속하기 때문에 오류를 반복 전파함
- Context Collapse: LLM을 사용한 end-to-end context rewrite의 근본적인 위험을 반영하며, 누적된 지식이 보존대는 대신 갑자기 지워질 수 있음

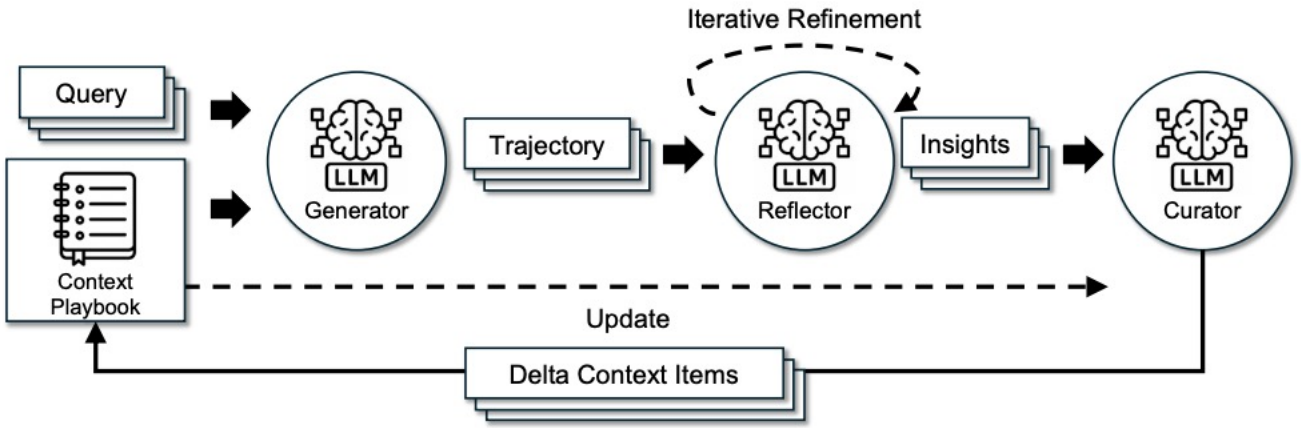
-> Agent Context는 요약 대상이 아니라, 전략과 경험을 보존하는 evolving playbook이어야 함.

Agentic Context Engineering (ACE)

Overview

- 기존 context adaptation의 두 가지 한계를 해결하기 위해 세 가지 혁신을 도입함

1. 평가 및 통찰력 추출을 큐레이션에서 분리하는 전용 Reflector를 통해 context 품질과 downstream 성능을 개선함
2. 비용이 많이 드는 단일체 재작성을 localized 편집으로 대체하는 incremental delta updates를 통해 latency와 computing cost 모두 줄임
3. 꾸준한 context 확장과 중복 제어를 균형 있게 유지하는 grow-and-refine 메커니즘임



-> 인간이 실험하고, 반성하고 통합하는 방식을 모방하여 단일 모델에 모든 책임을 부여하는 bottleneck을 피함

Agentic Context Engineering (ACE)

Incremental Delta Updates

- ACE는 context를 하나의 거대한 prompt가 아니라, 고유 ID와 유용성/위험성 counter를 가진 작은 bullet 단위로 관리함
- 각 bullet은 재사용 가능한 전략, 도메인 개념, 실패 패턴을 담으며, 필요한 bullet만 검색, 수정, 삭제 가능
- 전체 context를 다시 쓰지 않고, 새로 얻은 교훈만 작은 delta bullet으로 추가해 기존 지식을 보존하면서 점진적 확장

Grow-and-Refine

- 새로운 교훈은 새 ID를 가진 bullet으로 추가하고, 기존 bullet은 counter 증가처럼 필요한 부분만 제자리에서 업데이트함
- 의미론적 임베딩 기반 중복 제거를 통해 유사한 bullet을 병합하거나 제거하여, context가 불필요하게 비대해지는 것을 방지함

Results

Tasks and Datasets

- LLM Agent: APPWorld는 API 이해, 코드 생성 및 상호작용을 포함하는 자율 에이전트 태스크 모음으로 두 가지 난이도 수준의 태스크를 갖춤
- Domain-Specific Reasoning: 금융 분석을 위해 금융 추론 태스크에서 LLM을 테스트하는 FiNER (금융 문서 토큰 레이블링)와 Formula (금융 개념 적용 및 계산 수행) 에 초점을 맞춤. 이외에도 DDXPlus (의료 추론) 및 BIRD-SQL (test-to-SQL)에 대해 평가함

Evaluation Metrics

- AppWorld: Task Goal Completion 및 Scenario Goal Completion
- FiNER, Formula 및 DDXPlus: Accuracy, BIRD-SQL: GPT-4o 기반 LLM-as-a-judge 평가
- Offline context adaptation의 경우 pass@1 accuracy를, Online context adaptation의 경우 test set에서 순차 평가를 수행

Results

Baseline and Methods

- Base LLM and In-Context Learning Baselines
    1. Base LLM: ReAct 프레임워크에서 Default Prompt 사용
    2. In-Context Learning: 모델의 context window에 맞는 경우 모든 학습 샘플을, 초과하면 가능한 많은 shot 제공
- Prompt and Memory Optimization Baselines
    1. MIPROv2: Instruction과 demonstration을 Bayesian optimization으로 최적화
    2. GEPA: Execution trace와 Reflection을 활용해 prompt를 진화시킴
    3. Dynamic Cheatsheet: Reusable strategies와 code snippets를 외부 memory에 누적
- ACE Setting
    - Offline/Online context adaptation 모두에서 평가
    - Generator, Reflector, Curator 각각에 대해 동일한 DeepSeep-V3.1 사용

Results

Results on Agent Benchmark

Method	GT Labels	Test-Normal		Test-Challenge		Average
		TGC↑	SGC↑	TGC↑	SGC↑	
DeepSeek-V3.1-671B as Base LLM						
ReAct		63.7	42.9	41.5	21.6	42.4
Offline Adaptation						
ReAct + ICL	✓	64.3$_{+0.6}$	46.4$_{+3.5}$	46.0$_{+4.5}$	27.3$_{+5.7}$	46.0$_{+3.6}$
ReAct + GEPA	✓	64.9$_{+1.2}$	44.6$_{+1.7}$	46.0$_{+4.5}$	30.2$_{+8.6}$	46.4$_{+4.0}$
ReAct + ACE	✓	**76.2**$_{+12.5}$	**64.3**$_{+21.4}$	**57.3**$_{+15.8}$	**39.6**$_{+18.0}$	**59.4**$_{+17.0}$
ReAct + ACE	✗	75.0$_{+11.3}$	**64.3**$_{+21.4}$	54.4$_{+12.9}$	35.2$_{+13.6}$	57.2$_{+14.8}$
Online Adaptation						
ReAct + DC (CU)	✗	65.5$_{+1.8}$	**58.9**$_{+16.0}$	52.3$_{+10.8}$	30.8$_{+9.2}$	51.9$_{+9.5}$
ReAct + ACE	✗	**69.6**$_{+5.9}$	53.6$_{+10.7}$	**66.0**$_{+24.5}$	**48.9**$_{+27.3}$	**59.5**$_{+17.1}$

Results

Results on Domain-Specific Benchmark

Method	GT Labels	FiNER (Acc↑)	Formula (Acc↑)	Average
DeepSeek-V3.1 as Base LLM				
Base LLM		70.7	67.5	69.1
Offline Adaptation				
ICL	✓	72.3$_{+1.6}$	67.0$_{-0.5}$	69.6$_{+0.5}$
MIPROv2	✓	72.4$_{+1.7}$	69.5$_{+2.0}$	70.9$_{+1.8}$
GEPA	✓	73.5$_{+2.8}$	71.5$_{+4.0}$	72.5$_{+3.4}$
ACE	✓	**78.3**$_{+7.6}$	**85.5**$_{+18.0}$	**81.9**$_{+12.8}$
ACE	✗	71.1$_{+0.4}$	83.0$_{+15.5}$	77.1$_{+8.0}$
Online Adaptation				
DC (CU)	✓	74.2$_{+3.5}$	69.5$_{+2.0}$	71.8$_{+2.7}$
DC (CU)	✗	68.3$_{-2.4}$	62.5$_{-5.0}$	65.4$_{-3.7}$
ACE	✓	**76.7**$_{+6.0}$	76.5$_{+9.0}$	**76.6**$_{+7.5}$
ACE	✗	67.3$_{-3.4}$	**78.5**$_{+11.0}$	72.9$_{+3.8}$

Method	GT Labels	FiNER (Acc↑)	Formula (Acc↑)	Average
GPT-OSS-120B as Base LLM				
Base LLM		66.6	71.5	69.1
Offline Adaptation				
GEPA	✓	67.9$_{+1.3}$	71.5$_{+0.0}$	69.7$_{+0.6}$
ACE	✓	**73.8**$_{+7.2}$	**88.5**$_{+17.0}$	**81.2**$_{+12.1}$
ACE	✗	69.7$_{+3.1}$	84.0$_{+12.5}$	76.9$_{+7.8}$
Online Adaptation				
DC	✗	55.8$_{-10.8}$	66.0$_{-5.5}$	60.9$_{-8.2}$
ACE	✗	**70.5**$_{+3.9}$	**85.0**$_{+13.5}$	**77.8**$_{+8.7}$

Table 7: **Results on Financial Analysis Benchmark (GPT-OSS-120B as the Base LLM).** "GT labels" indicates whether ground-truth labels are available to the Reflector during adaptation.

Method	GT Labels	FiNER (Acc↑)	Formula (Acc↑)	Average
GPT-5.1 as Base LLM				
Base LLM		73.5	73.0	73.3
Offline Adaptation				
GEPA	✓	74.5$_{+1.0}$	73.0$_{+0.0}$	73.8$_{+0.5}$
ACE	✓	**81.0**$_{+7.5}$	**84.5**$_{+11.5}$	**82.8**$_{+9.5}$
ACE	✗	78.2$_{+4.7}$	76.5$_{+3.5}$	77.4$_{+4.1}$
Online Adaptation				
DC	✗	70.0$_{-3.5}$	69.0$_{-4.0}$	69.5$_{-3.8}$
ACE	✗	**75.2**$_{+1.7}$	**79.0**$_{+6.0}$	**77.1**$_{+3.8}$

Table 8: **Results on Financial Analysis Benchmark (GPT-5.1 as the Base LLM).** "GT labels" indicates whether ground-truth labels are available to the Reflector during adaptation.

Results

Ablation Study

Method	GT Labels	Test-Normal		Test-Challenge		Average
		TGC↑	SGC↑	TGC↑	SGC↑	
DeepSeek-V3.1 as Base LLM						
ReAct		63.7	42.9	41.5	21.6	42.4
Offline Adaptation						
ReAct + ACE w/o Reflector or multi-epoch	✓	70.8$_{+7.1}$	55.4$_{+12.5}$	55.9$_{+14.4}$	38.1$_{+17.5}$	55.1$_{+12.7}$
ReAct + ACE w/o multi-epoch	✓	72.0$_{+8.3}$	60.7$_{+17.8}$	54.9$_{+13.4}$	39.6$_{+18.0}$	56.8$_{+14.4}$
ReAct + ACE	✓	76.2$_{+12.5}$	64.3$_{+21.4}$	57.3$_{+15.8}$	39.6$_{+18.0}$	59.4$_{+17.0}$
Online Adaptation						
ReAct + ACE	✗	67.9$_{+4.2}$	51.8$_{+8.9}$	61.4$_{+19.9}$	43.2$_{+21.6}$	56.1$_{+13.7}$
ReAct + ACE + offline warmup	✗	69.6$_{+5.9}$	53.6$_{+10.7}$	66.0$_{+24.5}$	48.9$_{+27.3}$	59.5$_{+17.1}$

Results

Cost and Speed Analysis

Method	Latency (s)↓	# Rollouts↓
ReAct + GEPA	53898	1434
ReAct + ACE	9517 (-82.3%)	357 (-75.1%)

(a) **Offline** (AppWorld).

Method	Latency (s)↓	Token Cost ($)↓
DC (CU)	65104	17.7
ACE	5503 (-91.5%)	2.9 (-83.6%)

(b) **Online** (FiNER).

Discussion

Implications for Online and Continual Learning

- ACE는 모델 가중치를 업데이트하지 않고 context를 수정해 adapt하므로, fine-tuning보다 저비용, 고효율의 대안이 될 수 있음
- Context는 사람이 해석 가능하기 때문에, 오래되었거나 잘못된 정보, 또는 privacy/legal issue가 있는 정보를 선택적으로 제거하는 selective unlearning이 가능함
- 따라서 ACE는 distribution shift, 제한된 training data, continual learning 상황에서 유연한 context-level adaptation 방법으로 활용될 수 있음

Limitations and Challenges

- ACE는 Reflector가 실행 trace와 결과에서 의미 있는 insight를 추출할 수 있어야 효과적임
- 모든 task가 풍부한 context를 필요로 하는 것은 아님 (ex: HotPotQA)
- ACE는 세부 도메인 지식, 복잡한 tool use, 환경 특화 전략이 필요한 agentic/domain-specific task에서 가장 효과적임

AGENTFOLD: LONG-HORIZON WEB AGENTS WITH PROACTIVE CONTEXT FOLDING

Rui Ye[1,2,*], **Zhongwang Zhang**[1,2,*], **Kuan Li**[2,*], **Huifeng Yin**[2,*]
Zhengwei Tao[2], **Yida Zhao**[2], **Liangcai Su**[2], **Liwen Zhang**[2], **Wenbiao Yin**[2], **Zile Qiao**[2]
Xinyu Wang[2], **Pengjun Xie**[2], **Fei Huang**[2], **Jingren Zhou**[2], **Siheng Chen**[1,#], **Yong Jiang**[2,#]
[1]Shanghai Jiao Tong University, [2]Alibaba Tongyi Lab
[*]Equal Contribution, [#]Corresponding Author (sihengc@sjtu.edu.cn, jiangyong.ml@gmail.com)

ICLR 2026

Introduction

Context Management Challenge in Web Agents

- ReAct 기반 agent: reasoning-action-observation 전체 기록을 유지해 정보는 보존하지만, raw web data의 noise가 누적됨
- Step-wise summarization: context는 간결하게 유지하지만, 한 번의 요약 과정에서 중요한 세부 정보가 손실될 수 있음

-> 정보를 충분히 보존하면서도 불필요한 noise를 줄이는 context 관리 방식이 필요함

AgentFold: Proactive Context Folding

- Agent의 context를 단순히 누적되는 로그가 아니라, 사람이 사용하는 mental scratchpad처럼 능동적으로 관리함
- Context 구성:
  1. Multi-Scale State Summaries - 과거 사건을 여러 수준으로 압축한 요약 기록
  2. Latest Interaction – 가장 최근 action과 observation의 완전한 기록
- Folding 방식:
  1. Granular condensation – 최신 interaction을 새로운 state summary로 압축
  2. Deep consolidation – 여러 summary와 최신 interaction을 더 높은 수준의 abstraction으로 통합

Related Work

Web Agents

- WebThinker, WebDancer, WebSailor, WebExplorer 등은 주로 Web task dataset construction과 test-time scaling에 초점을 둠
- 하지만, ReAct의 append-only context는 long-horizon task에서 context saturation을 유발하며, 중요한 정보가 noise 속에 묻힐 수 있음

Context Management

- Context management는 LLM agent에게 적절하고 효과적인 context를 제공하기 위한 연구 흐름임
- 기존 연구는 외부 지식, 사용자 profile, 과거 대화 등을 주입하는 External Context Augmentation에 주로 집중함
- AgentFold는 task 내부에서 생성된 context를 관리하는 Intra-Task Context Curation에 초점을 두며, rigid step-wise summarization 대신 flexible look-back folding을 제안함

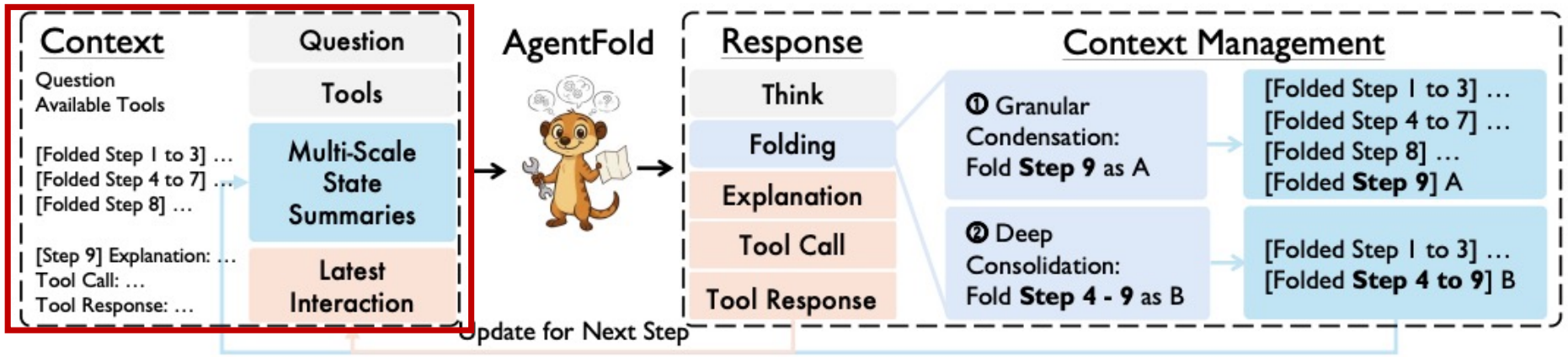
AgentFold: Web Agent with Proactive Context Folding

Overview

1. 에이전트의 context를 단일한 log가 아닌 동적인 인지 작업 공간으로 정의함
2. 에이전트가 추론 과정의 본질적인 부분으로서 이 작업공간을 능동적으로 조작하고 조각할 수 있도록 함

- 일반적인 단계에서 에이전트의 추론은 과거 상태 요약을 관리하기 위한 folding directive, thought 과정에 대한 설명, 그리고 다음 action으로 구성된 multi-part response를 생성함
- Folding directive는 미래 step을 위한 multi-scale state summary를 위해 즉시 적용되고, reasoning, action, observation은 다음 주기를 위한 latest interaction을 형성함
- 이 운영 주기는 curation이 수동적인 부산물이 아닌 명시적이고 학습된 단계인 강력한 perceive -> reason -> fold -> action 루프를 설정함
- 잘 인지된 작업 공간과 이를 조작하는 에이전트의 자율성 효과를 시너지 효과를 통해 결합함으로써, AgentFold는 세부 정보 유지와 context 팽창 방지 사이의 중요한 절충점을 직접적으로 해결함

AgentFold's Context: Multi-Scale State Summaries, Latest Interaction



Multi-Scale State Summaries

- 중요한 발견은 find-grained summary로 유지
- 덜 중요한 중간 과정은 더 큰 abstract block으로 통합
- 전체 history의 논리적 흐름은 유지하면서 noise를 줄임

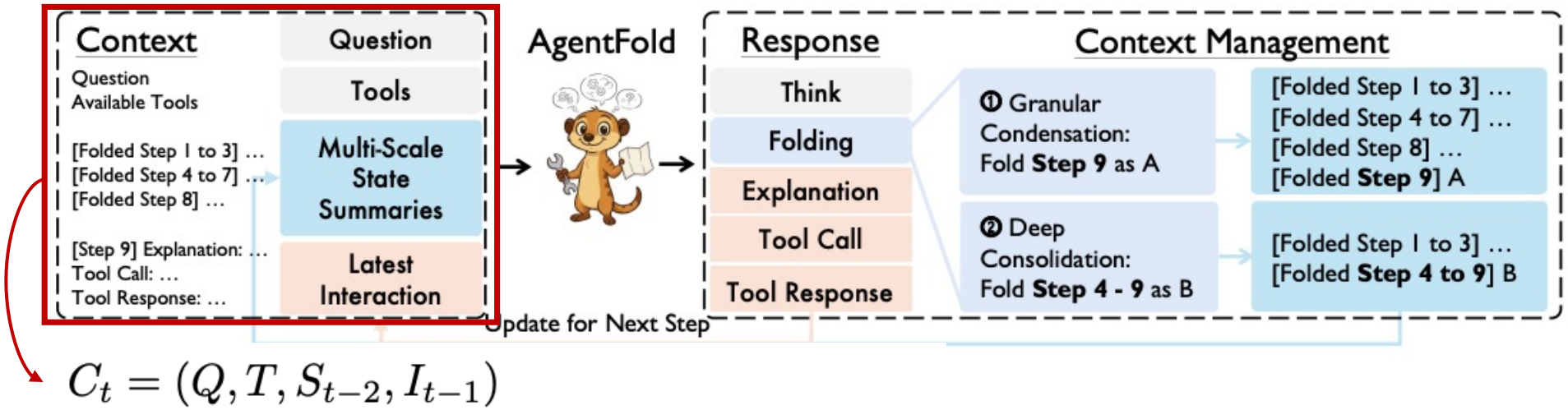
Latest Interaction

- 가장 최근 step의 세부 정보를 손실 없이 제공
- 다음 action을 결정하고, 어떤 정보를 folding할지 판단하는 데 필요한 situational awareness 제공

-> 전체 아키텍처는 인간이 안정적인 목표, 통합된 지식, 휘발성 작업 기억을 활용하는 방식을 반영함

AgentFold: Web Agent with Proactive Context Folding

AgentFold's Context: Multi-Scale State Summaries, Latest Interaction



$$C_t = (Q, T, S_{t-2}, I_{t-1})$$

$$S_t = (s_{x_1,y_1}, s_{x_2,y_2}, \ldots, s_{x_m,y_m})$$

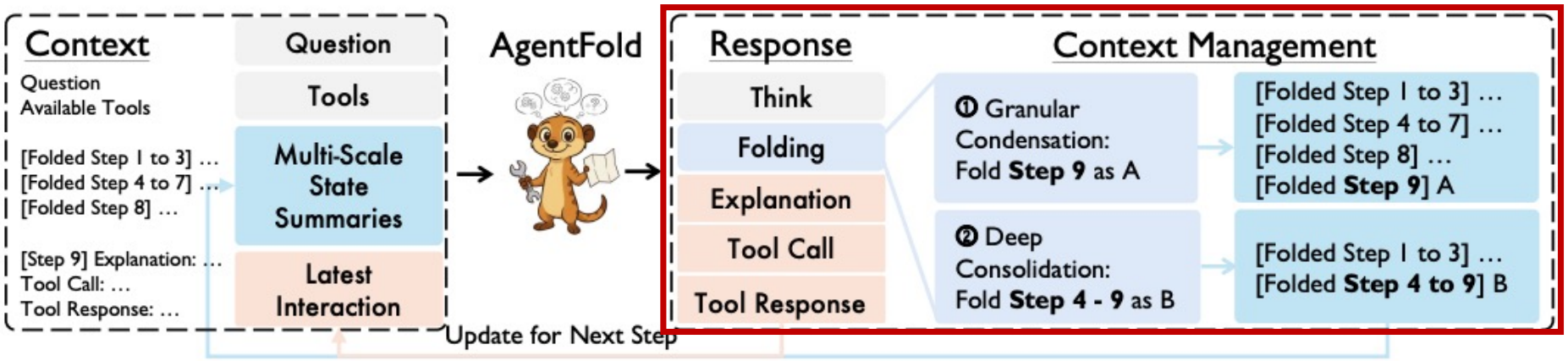
- $S_{x,y}$: step x부터 y까지를 요약한 summary block
- $S_{x,x}$: 단일 step에 대한 fine-grained summary
- $S_{x,y}, y > x$: 여러 step을 통합한 coarse-grained summary

$$I_{t-1} = (e_{t-1}, a_{t-1}, o_{t-1})$$

- e_{t-1}: explanation
- a_{t-1}: action
- o_{t-1}: observation

AgentFold: Web Agent with Proactive Context Folding

AgentFold's Response: Thinking, Folding, Explanation, Action



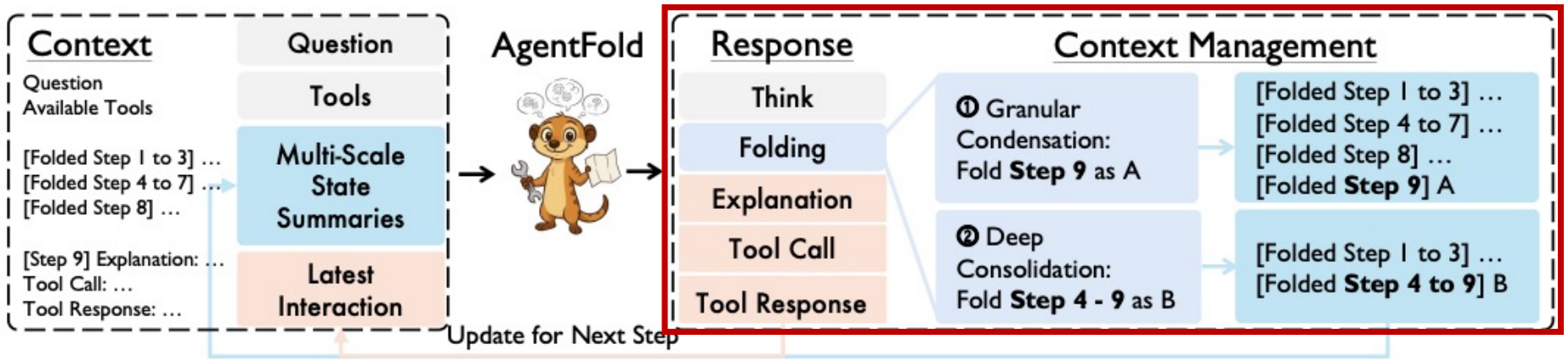
- AgentFold의 response는 단일 action command가 아니라, context 관리와 task 수행을 함께 포함하는 multi-part output임

$$R_t = \text{AgentFold}(C_t; \theta) \rightarrow (th_t, f_t, e_t, a_t)$$

- th_t: 현재 context를 분석하고 folding 및 다음 action을 판단하는 thinking process
- f_t: 과거 state summaries를 어떻게 fold할지 결정하는 folding directive
- e_t: 선택한 action의 이유를 설명하는 concise explanation
- a_t: 다음 tool call 또는 최종 답변

AgentFold: Web Agent with Proactive Context Folding

AgentFold's Response: Thinking, Folding, Explanation, Action



- Folding directive는 다음 형태의 JSON object로 표현됨: $f_t = \{\text{“range”} : [\text{k,t-1}], \text{“summary”} : \text{“}\sigma_t\text{”}\}$
- 두 가지 folding mode
  1. Granular Condensation: K = t-1 <- Latest Interaction 하나만 fine-grained summary로 압축
  2. Deep Consolidation: K < t-1 <- 여러 prior summaries와 Latent Interaction을 하나의 coarse-grained summary로 통합
- Context update: range [k, t-1]에 해당하는 summary blocks를 제거 후 새 summary block으로 대체

Discussions

- Folding이 특정 range에만 적용되어 historical prefix 상당 부분이 유지되어 해당 prefix는 KV cache 재사용이 가능함
- 즉, full-history summarization 처럼 매번 전체 context를 다시 써서 cache를 무효화하는 비용을 줄임

AgentFold's Training Data Trajectory Collection

- AgentFold는 action selection과 context folding이 함께 포함된 structured trajectory가 필요함
- WebSailor와 동일한 question set 사용하여 강력한 open-source LLM (DeepSeek-V3.1, GLM-4.5)으로 (Context, Response) interaction pairs 생성
- Prompt engineering만으로는 정확한 multi-part response 생성이 어려워 rejection sampling 적용 -> 최종적으로 validated gold-standard response만 학습 데이터로 사용
- Curated dataset으로 open-source LLM (Qwen3-30B-A3B-Instruct)을 SFT하여 folding 능력을 prompt-dependent instruction이 아니라 모델 내부 skill로 학습

Experiments

Implementation

- Base model: Qwen3-30B-A3B-Instruct-2507 (Prediction 시 3B activate)
- Max tool calls = 100
- Fold Generator에는 GLM-4.5와 DeepSeek-V3.1 사용

Benchmarks

- Imformation-seeking benchmarks: BrowseComp, BrowseComp-ZH, WideSearch-en
- General benchmark: GAIA text-only subset

Baselines

- Open-source agents: WebThinker, WebDancer, WebSailor, WebExplorer, DeepSeek-V3.1, GLM-4.5 등
- 상용 agents: Claude-4-Sonnet/Opus, OpenAI o4-mini/o3, OpenAI Deep Research 등

Experiments

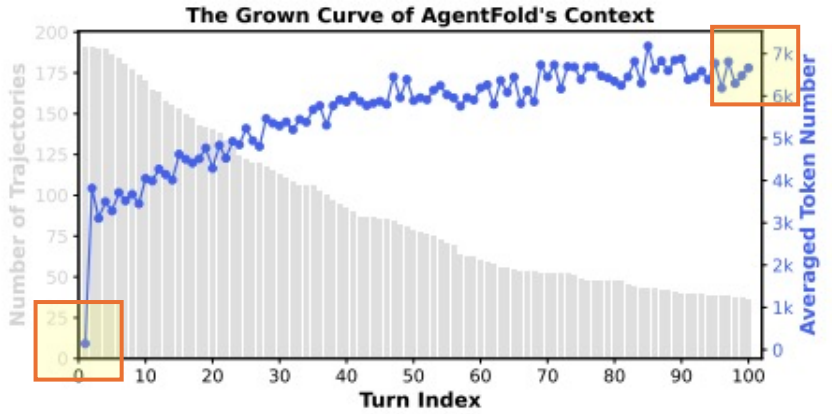
Results and Analysis – Main Results

Agent	BrowseComp	BrowseComp-ZH	WideSearch	GAIA
Proprietary Agents				
Claude-4-Sonnet	14.7	22.5	**62.0**	68.3
Claude-4-Opus (anthropic, 2025)	18.8	37.4	-	-
OpenAI-o4-mini (OpenAI, 2025b)	28.3	44.3	-	-
OpenAI-o3 (OpenAI, 2025b)	49.7	**58.1**	60.0	**70.5**
OpenAI Deep Research (OpenAI, 2025a)	**51.5**	42.9	-	67.4
Open-Source Agents				
Qwen3-30B-A3B Yang et al. (2025)	0.5	13.5	-	35.9
WebThinker-32B Li et al. (2025d)	2.8	7.3	-	48.5
WebDancer-32B (Wu et al., 2025)	3.8	18.0	-	51.5
WebSailor-32B (Li et al., 2025b)	10.5	25.5	-	53.2
WebSailor-72B (Li et al., 2025b)	12.0	30.1	-	55.4
ASearcher-Web-32B (Gao et al., 2025)	5.2	15.6	-	52.8
MiroThinker-32B-DPO-v0.2 (MiroMind AI Team, 2025)	13.0	17.0	-	64.1
WebExplorer-8B (Liu et al., 2025)	15.7	32.0	-	50.0
DeepDive-32B (Lu et al., 2025)	14.8	25.6	-	-
DeepDiver-V2-38B (OpenPangu Team, 2025)	13.4	34.6	-	-
Kimi-K2-Instruct-1T (Team et al., 2025)	14.1	28.8	59.9	57.3
GLM-4.5-355B-A32B (Zeng et al., 2025)	26.4	37.5	-	66.0
DeepSeek-V3.1-671B-A37B (DeepSeek Team, 2025)	30.0	**49.2**	-	63.1
AgentFold-30B-A3B (Ours)	**36.2**	47.3	**62.1**	**67.0**

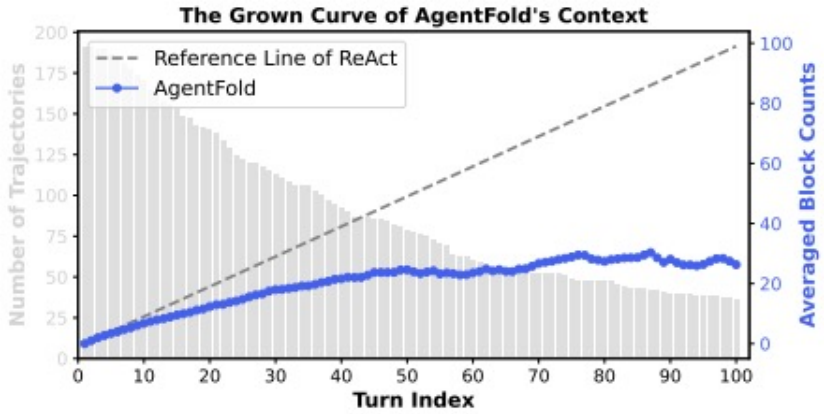
- 장기 정보 탐색 task에서는 효과적인 context management가 훨씬 큰 모델과의 성능 격차를 줄일 수 있다는 점을 보여줌

Experiments

Results and Analysis – Dynamics of AgentFold's Context: token and block count



(a) Growth curve of AgentFold's context

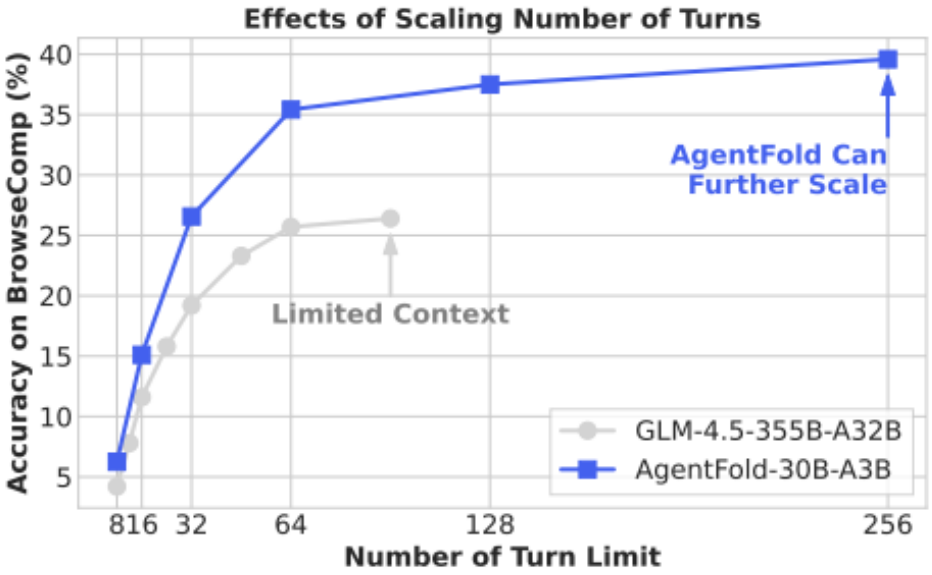


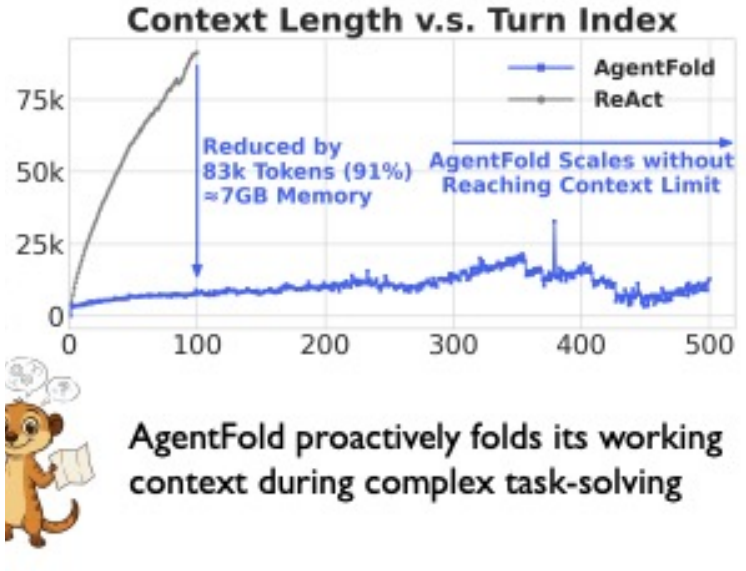
(b) Number of blocks in AgentFold's context

- 장기 정보 탐색 task에서는 효과적인 context management가 훨씬 큰 모델과의 성능 격차를 줄일 수 있다는 점을 보여줌

Experiments

Results and Analysis – Scaling Properties of Interaction Turns





- 단순히 모델 크기보다 context를 어떻게 관리하는지가 long-horizon task에서 중요하다는 점을 보여줌
- Append-only context가 길어지면 기존 agent는 noise와 context saturation에 취약해지지만, AgentFold는 folding을 통해 더 긴 interaction을 효과적으로 활용 가능함

Conclusion

Key Contributions

- ReAct의 append-only context saturation과 full-history summarization의 정보 손실 문제 사이의 trade-off를 해결하려는 새로운 agent paradigm 제안
- Granular Condensation과 Deep Consolidation으로 Proactive Context Fold를 실현함
- Compact한 context를 유지하면서 수백 단계의 long-horizon interaction 가능성을 제시함

Future Direction

- 현재는 단순 SFT 기반으로 가능성을 입증함
- 향후 RL을 통해 task success를 직접 최적화하고, 더 정교한 folding policy를 학습할 수 있음

Thank You