Language Models can Solve Computer Tasks

Geunwoo Kim[1], Pierre Baldi[1], Stephen McAleer[2]

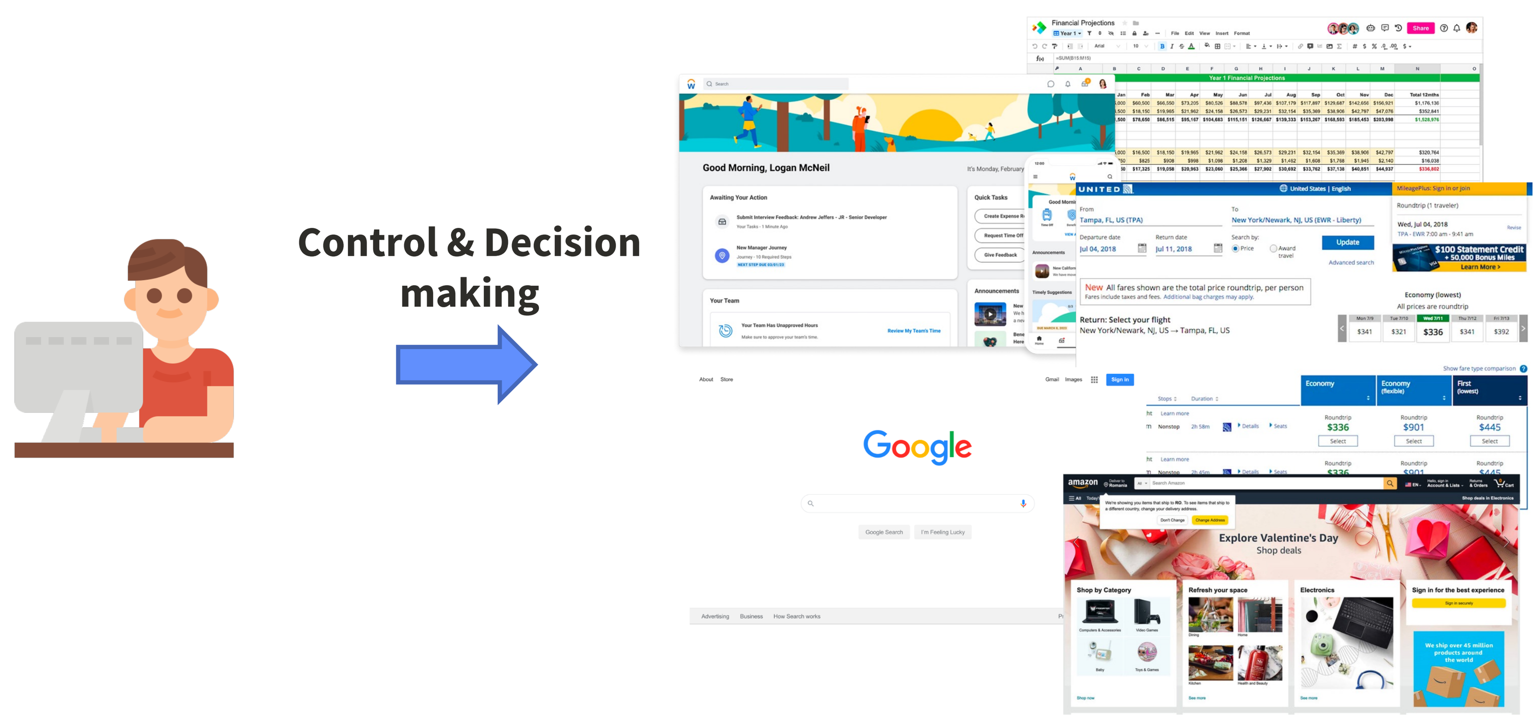
[1]*University of California, Irvine*, [2]*Carnegie Mellon University*

Summary

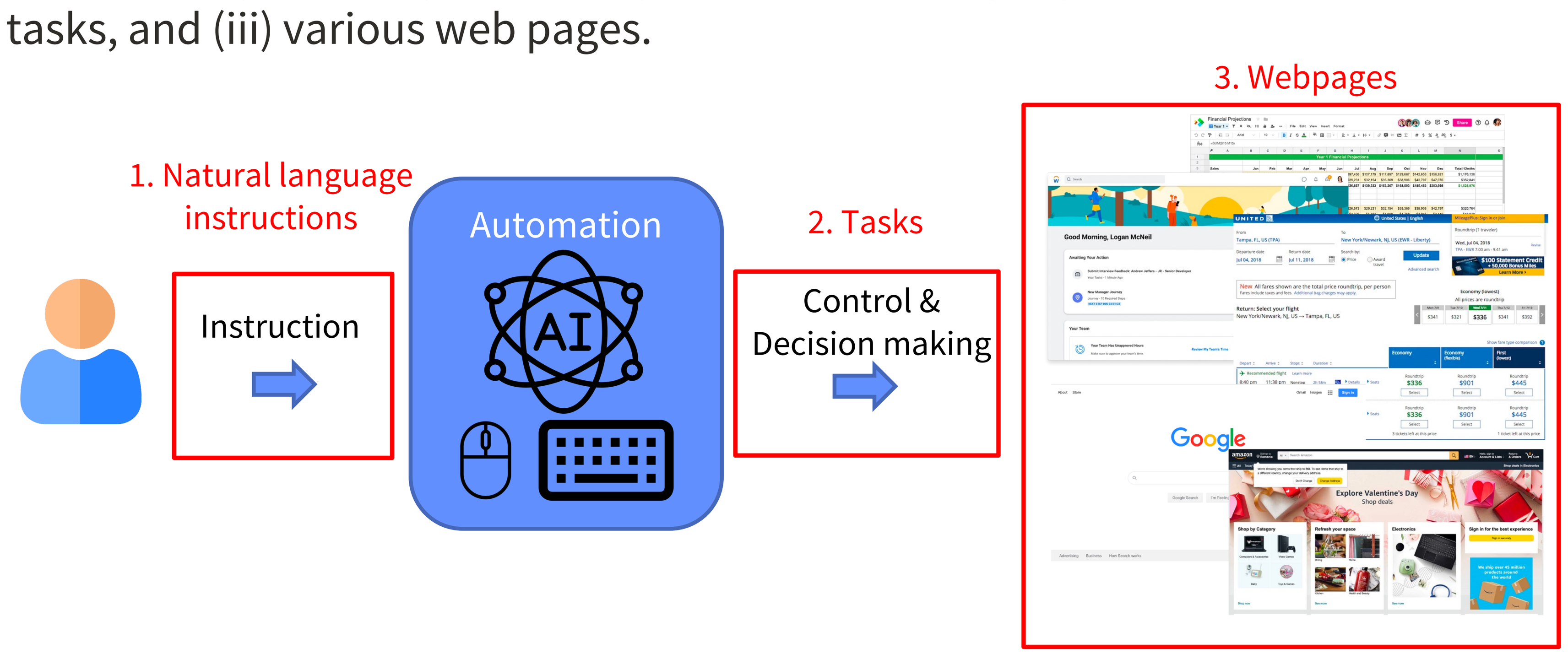
This research presents a novel approach for automating computer tasks using a pre-trained large language model (LLM) agent guided by natural language prompts. The proposed prompting method, Recursively Criticize and Improve (RCI), outperforms existing LLM methods, supervised learning (SL), and reinforcement learning (RL) approaches on the MiniWoB++ benchmark. Additionally, RCI prompting is shown to enhance LLMs' reasoning abilities on various natural language reasoning tasks, surpassing the chain of thought (CoT) prompting approach.

Motivation

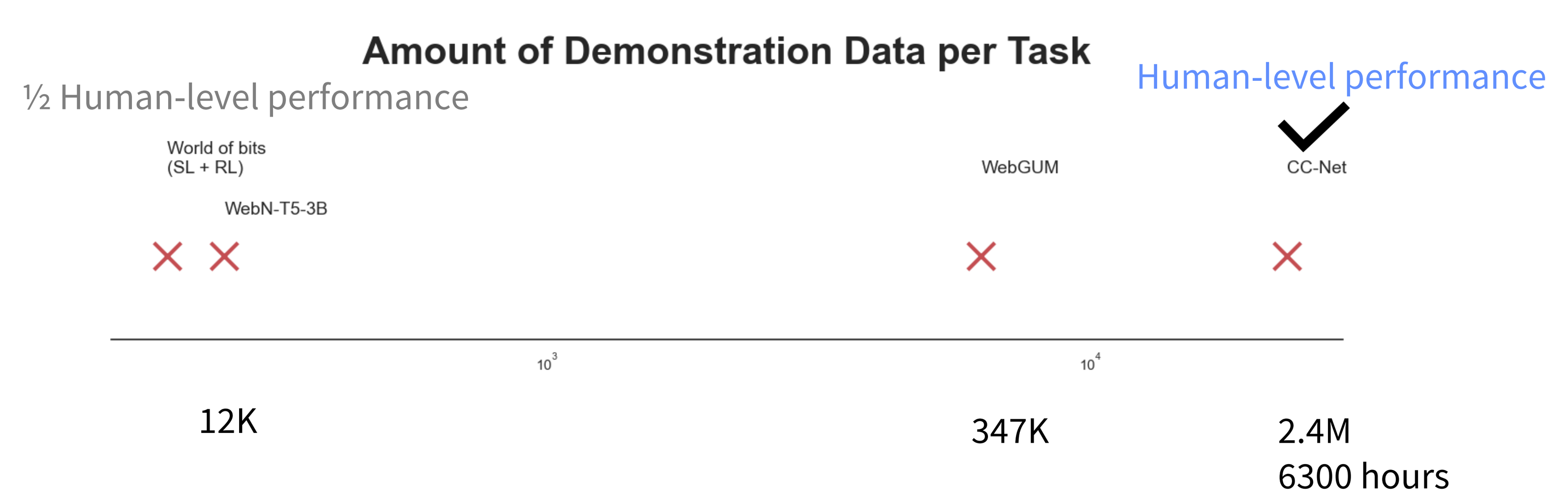
The goal is to develop an automatic agent capable of controlling a computer and performing tasks typically carried out by humans. This agent aims to achieve a level of cognitive ability comparable to humans, enabling it to solve a wide range of computer tasks through natural language communication.



The challenges lie in generalizing to (i) natural language instructions, (ii) diverse tasks, and (iii) various web pages.



The limitation of previous approaches (Imitation learning and reinforcement learning) is that they require large amounts of expert demonstrations and task-specific reward functions, which are impractical.



Our solution: LLMs

World knowledge stored in the foundation model

In-context learning for a few-shot approach

We got:

No reward function is needed!

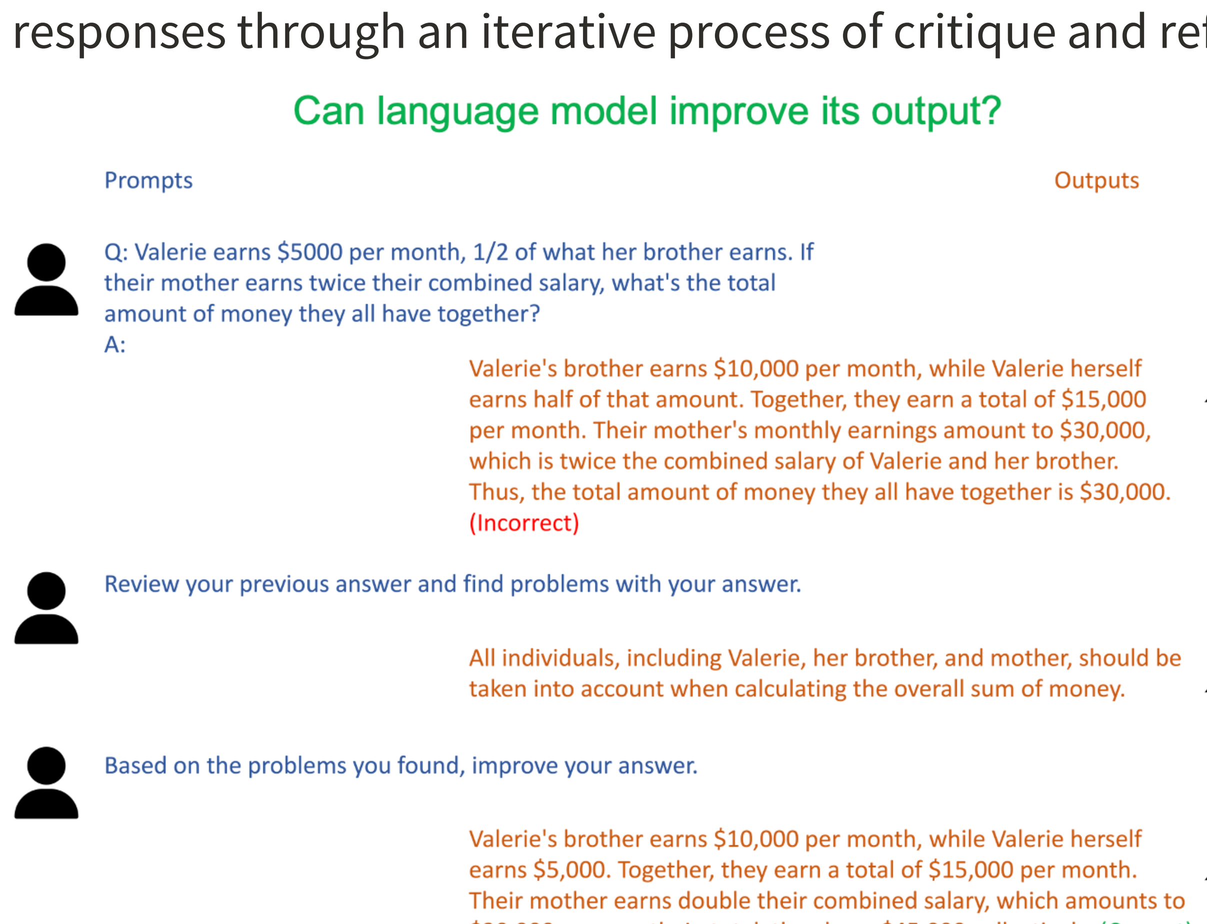
Generalization to

(i) instructions (ii) tasks (iii) webpages

Less than 2~3 demonstration per task

RCI prompting

To ground LLMs on computer tasks, we propose RCI prompting where we prompts the models to identify and correct errors in their output, leading to improved responses through an iterative process of critique and refinement.



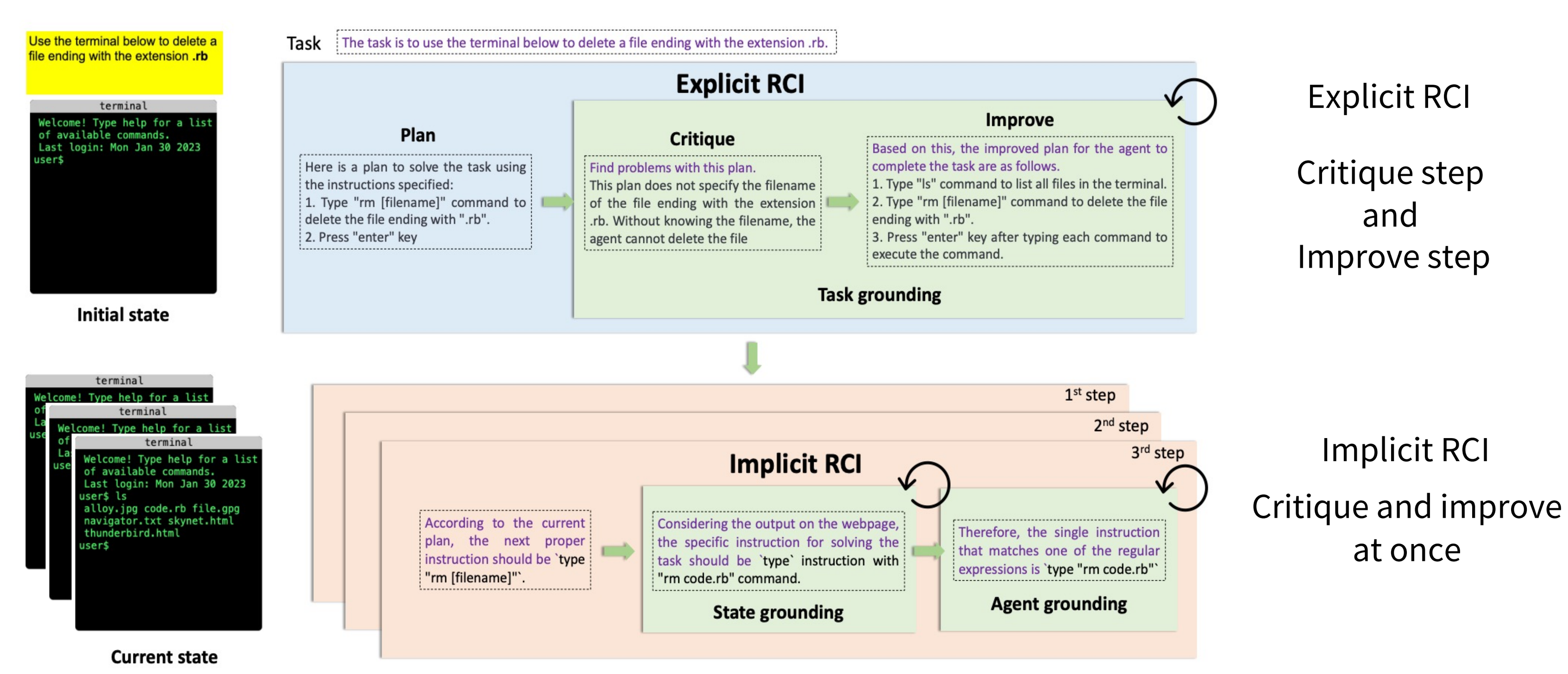
Evaluation on reasoning tasks

The performance comparison between RCI and CoT baselines on arithmetic reasoning tasks shows that Zero-Shot + RCI outperforms Zero-Shot CoT and Few-Shot CoT in most tasks. RCI prompting demonstrates a collaborative impact when combined with CoT baselines, leading to the highest scores in four out of five tasks and suggesting potential for future research in combining RCI with other prompting methods.

	GSM8K	MultiArith	AddSub	SVAMP	SingleEq
Zero-Shot	78.35	96.06	85.83	78.35	91.34
Zero-Shot + RCI	85.43	97.64	89.76	84.65	**94.49**
Zero-Shot CoT	82.28	96.85	83.86	79.92	89.37
Zero-Shot CoT + RCI	**86.22**	97.24	89.88	85.83	90.94
Few-Shot CoT	80.31	98.82	89.37	83.46	91.73
Few-Shot CoT + RCI	84.25	**99.21**	**90.55**	**87.40**	93.70

RCI for Computer Tasks

To ensure the output of LLMs is grounded on task, state, and agent, we employ RCI prompting.



Explicit RCI

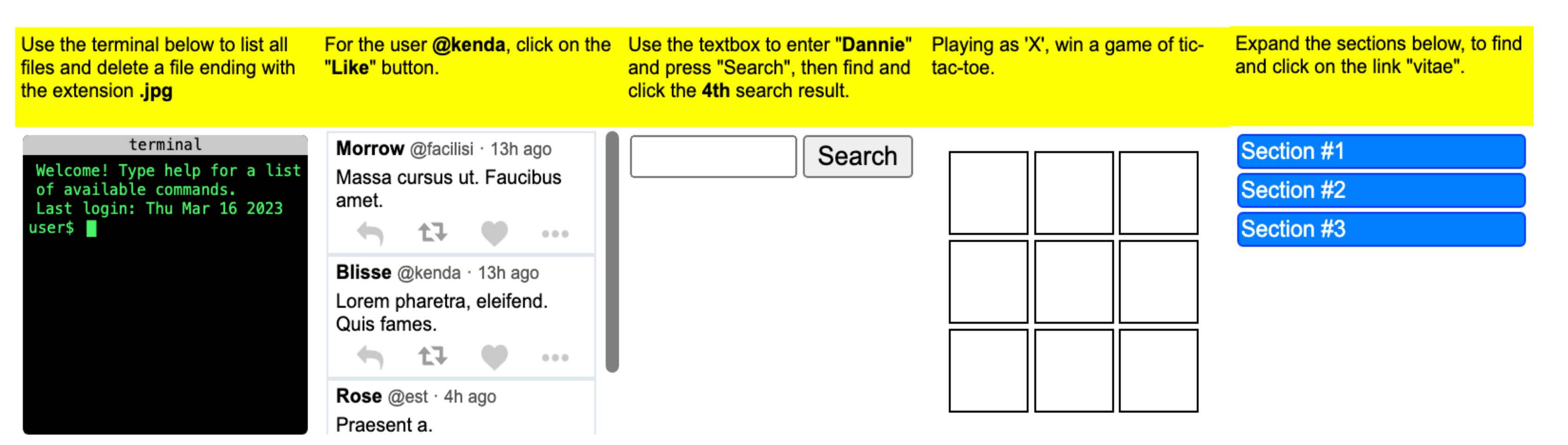
Critique step and Improve step

Implicit RCI

Critique and improve at once

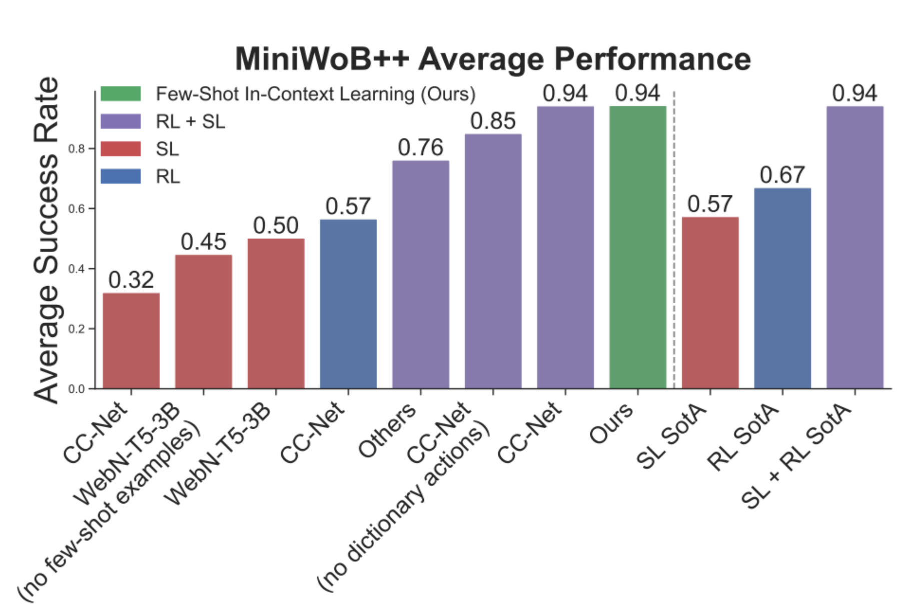
MiniWoB++ benchmark

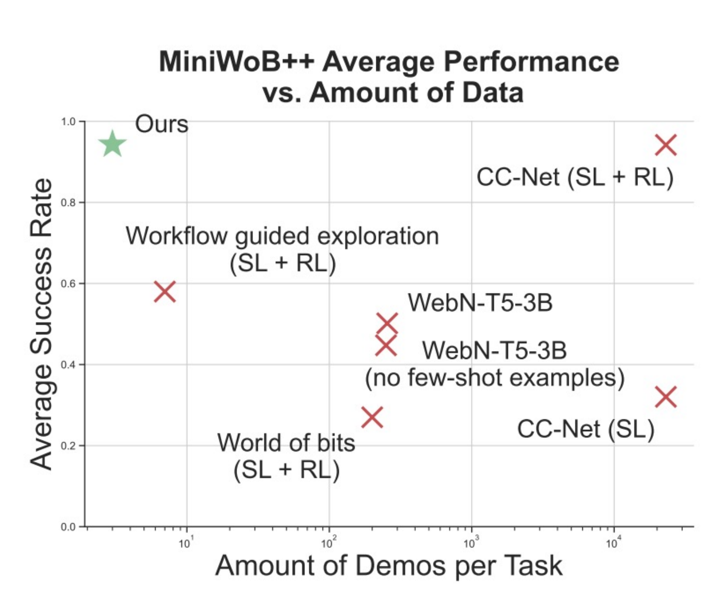
We use MiniWoB++ benchmark which is a web-based simulation environment designed to evaluate computer agents. It offers a diverse range of tasks, from simple button-clicking to complex compositional tasks, with a shared action space involving keyboard and mouse interactions and a state space centered around HTML code.



Evaluation on MiniWoB++

The RCI agent achieves state-of-the-art on MiniWoB++ with the lowest sample complexity, requiring 120 times fewer samples than *WebN-T5-3B* and 11,000 times fewer samples than *CC-Net*.





Next steps

- Explore the use of multimodal foundation models for state representation.
- Investigate the performance of fine-tuned LLMs on computer tasks.
- Optimize the performance by exploring the space of potential loop and prompt structures and identifying the most promising ones.
- Develop advanced variations of RCI to build upon the basic framework and enhance its performance.

GitHub

Website