Grounded Language Agent for Product Search via Intelligent Web Interactions

Moghis Fereidouni, Adib Mosharrof, A.B. Siddique
University of Kentucky, Lexington, KY, USA
moghis.fereidouni@uky.edu, amo304@g.uky.edu, siddique@cs.uky.edu

Abstract

The development of agents powered by large language models (LLMs) to accomplish complex high-level user intents, has attracted significant attention recently. However, employing LLMs with billions of parameters (e.g., GPT-4) may incur substantial costs on top of handcrafting extensive prompts. To address this, we introduce a Grounded Language Agent for Intelligent Web Interactions, named GLAINTEL. GLAINTEL employs Flan-T5 as its backbone and is flexible in training in various settings: unsupervised learning, supervised learning, and unsupervised domain adaptation. Specifically, we tackle both the challenge of learning without human demonstrations and the opportunity to leverage human demonstrations effectively when those are available. Additionally, we explore unsupervised domain adaptation for cases where demonstrations are limited to a specific domain. Experimental evaluations across diverse setups demonstrate the effectiveness of GLAINTEL in unsupervised settings, outperforming in-context learning-based approaches that employ larger models with up to 540 billion parameters. Surprisingly, behavioral cloning-based methods that straightforwardly use human demonstrations do not outperform unsupervised variants of GLAINTEL. Additionally, we show that combining human demonstrations with reinforcement learning-based training yields results comparable to methods utilizing GPT-4. The code is available at: https://github.com/MultifacetedNLP/Web-Agents-Unsupervised.

1 Introduction

Large Language Models (LLMs) have demonstrated their proficiency in diverse tasks such as text classification, information extraction, and question answering (Bommasani et al., 2021; Brown et al., 2020; Vaswani et al., 2017; Raffel et al., 2020; Radford et al., 2019). Similarly, reinforcement learning (RL) has evolved as a powerful paradigm for training intelligent agents to navigate complex environments (Huang et al., 2022b; Ahn et al., 2022; Liang et al., 2023). Moreover, recent research highlights the capabilities of agents powered by LLMs. For example, agents utilizing GPT-4 can explore the virtual world in Minecraft, acquire a diverse set of composable skills, and exhibit exceptional proficiency in playing the game (Wang et al., 2024). The exceptional amount of world knowledge, often derived from vast text datasets, opens up possibilities for developing LLM-assisted intelligent web navigation agents capable of navigating and interacting with web pages akin to humans.

Despite their remarkable capabilities, off-the-shelf pre-trained LLMs face challenges in grounding and aligning themselves in interactive web environments (Mahowald et al., 2023). This limitation hampers their functional competence without additional customization. Additionally, employing LLMs with billion-scale parameters, such as GPT-4, may incur substantial costs on top of handcrafting extensive prompts. On the other hand, training smaller LLMs (e.g., Flan-T5) as agents can be challenging. For instance, consider a real-world product search scenario, where effective query formulation requires the agent to operate over a huge action space (i.e., language vocabulary), and navigating through diverse web pages poses additional challenges that need strategic exploration due to the presence of different actions on each page (i.e., dynamic action space). This complexity prevents the straightforward utilization of an action head on top of LLM. Moreover, the challenge extends to preserving long-term memory capabilities, which are crucial for comparing items or backtracking during the search process.

In this work, we introduce GLAINTEL, a **G**rounded **L**anguage **A**gent designed for **Intel**ligent Web Interactions. Given a user's intent specifying a product requirement, GLAINTEL formulates queries, navigates diverse web pages, and

Proceedings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U), pages 63–75

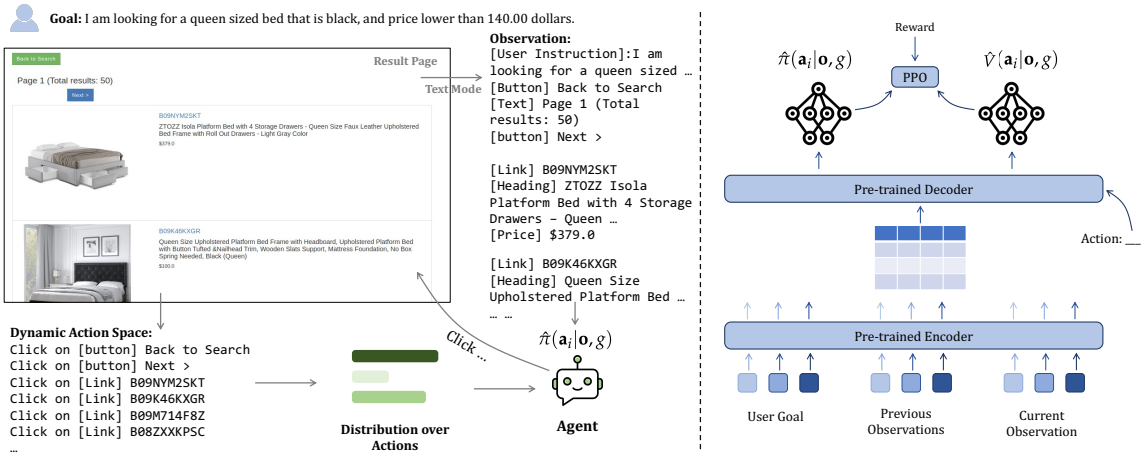

Figure 1: Overview of GLAINTEL: Our agent employs the Flan-T5 architecture and incorporates a language modeling head to adapt to dynamic action space, while the value head enables precise value estimation.

executes various actions to identify, customize, and purchase the desired product. GLAINTEL uses the open-source Flan-T5 language model (i.e., 780 million parameters) as its backbone and can be flexibly trained in various scenarios: unsupervised, supervised, and unsupervised domain adaptation settings. Specifically, we address the following research questions.

- *RQ1: Effectiveness of Unsupervised Learning:* Can LLM-based agents learn to address effective query generation and exploration of complex web pages with no human demonstrations?
- *RQ2: Impact of Human Demonstrations:* Can incorporating human demonstrations facilitate LLM-based agents to improve their overall performance? How to effectively leverage human demonstrations for training robust agents?
- *RQ3: Unsupervised Domain Adaptation:* Can LLM-based agents generalize to new, unseen product categories where no human demonstrations are available?

We employ a language modeling head to accommodate a *dynamic action space* and introduce an additional value head for precise value estimates. Figure 1 provides an overview of GLAINTEL. The user's goal and observation are sequentially passed to the model at each step. First, we obtain the input representation for every potential action token and compute the normalized joint probability for each action conditioned on the user goal and observation. Following the estimation of each action's probability, we apply a softmax function over these probabilities and sample an action according to this distribution. We fine-tune the agent using the Proximal Policy Optimization (PPO) algorithm (Dhariwal et al., 2017).

We conduct extensive experimental evaluations across diverse setups using the WebShop environment (Yao et al., 2022). WebShop is a simulated yet realistic e-commerce web platform featuring 1.18 million real-world products and 12,087 crowdsourced natural language intents. Based on our empirical study, we demonstrate that training Flan-T5 (e.g., 780 million parameters) in the unsupervised setting (i.e., no human demonstrations) can outperform in-context learning methods (Sridhar et al., 2023) that rely on models with up to 540 billion parameters. To quantify the impact of human supervision, we utilized 1010 human demonstrations for training supervised learning models using behavior cloning (BC) (Pomerleau, 1988).

Our findings indicate that incorporating human demonstrations through *straightforward BC does not produce superior results* when compared to the unsupervised RL-based PPO algorithm. Furthermore, our investigations reveal that leveraging human demonstrations through BC and then further training the agent with PPO in the unsupervised setting leads to the best results. Remarkably, this approach achieves results comparable to the method (Ma et al., 2023) that utilizes GPT-4. In the unsupervised domain adaptation (UDA) experiment, we observe that incorporating human demonstrations from a single category enables the agent to generalize to new product categories where no human demonstrations are available. Additionally, we *evaluate our trained model on a real website eBay* without any additional fine-tuning, which shows comparable results to methods that use the state-of-the-art GPT-4 model.

2 Proposed Agent: GLAINTEL

2.1 Problem Formulation

Given a user intent in natural language, the agent's goal is to buy the most appropriate product that fulfills the user's intent. We formulate the task as a goal-augmented Partially Observable Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{G}, \mathcal{O}, \gamma)$, where $\mathcal{S}$ is a set of states $\mathbf{s} \in \mathcal{S}$; $\mathcal{A} \subset \mathcal{V}^N$ represents action space sampled from LLM's vocabulary $\mathcal{V}$ of size N; $\mathcal{G} \subset \mathcal{V}^N$ denotes the goal space; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition function; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \mapsto \mathbb{R}$ characterizes the goal-conditioned reward function; $\mathcal{O}$ is a set of observations $\mathbf{o} \in \mathcal{O}$ (i.e., web page visible to agent); γ is the discount factor. We employ the language modeling head (i.e., distribution over the vocabulary) to accommodate the dynamic action space, which also facilitates directly computing the log probabilities of each action $\mathbf{a}_i = (w_0, \cdots, w_{|\mathbf{a}_i|})$ sampled from a dynamic action space given the agent's goal $g \in \mathcal{G}$ and observation $\mathbf{o}$.

It is important to note that each observation (i.e., web page) presents a dynamic set of actions to the agent, which prevents us from learning a probability distribution over the action space as in classification tasks. For instance, a search page allows actions such as typing an open-ended textual query or pressing the 'Search' button. Conversely, a product detail page offers actions such as 'Back to Search', '< Prev', 'Description', 'Features', 'Reviews', 'Buy Now', and the product-specific variable number of options. Figure 1 shows the observation and action space for the 'search result page'.

2.2 Overview of GLAINTEL

We employ Flan-T5 [1] as the core architecture, with the integration of the language modeling head and value head on top of the model. Our proposed agent, GLAINTEL, is adaptable to training across various setups: (*i*) unsupervised learning: no human demonstrations are available; (*ii*) unsupervised domain adaptation: limited human demonstrations in a single domain are available; and (*iii*) supervised learning: human demonstrations are accessible. In the following, we detail the specifics of the training and inference phases. The inclusion or exclusion of these phases is contingent upon the availability of the human demonstration data.

[1] Checkpoints: https://github.com/google-research/t5x/blob/main/docs/models.md#flan-t5-checkpoints

2.3 Optional Phase One: Supervised Training

The human demonstrations can serve as mappings from states to actions. Techniques such as imitation learning or behavioral cloning (BC) (Pomerleau, 1988) can be employed to fine-tune the policy π by minimizing the following loss over a dataset $\mathcal{D}$ comprising human demonstrations:

$$\mathcal{L}(\pi) = \mathbb{E}_{(s,a)\sim\mathcal{D}}[-\log \pi(a|s)].$$

The above formulation can be adapted to incorporate the interaction history with web pages $\pi(\mathbf{a}_t|\mathbf{s}_t, \tau_{<t})$, where $\tau_{<t}$ refers to the interaction trajectory leading up to time t. Subsequently, this formulation readily extends to utilize LLMs to learn an optimal policy where the encoder encodes the history of observations $(\mathbf{s}_t, \tau_{<t})$ and the decoder generates the next action $\mathbf{a}_t$ as:

$$\mathcal{L}_{\text{LLM}}(\pi) = \mathbb{E}_{\tau\sim\mathcal{D}}[\sum_{t=0}^{L} -\log \pi(\mathbf{a}_t|\tau_{<t}, \mathbf{s}_t)].$$

Building upon the recent works in return-conditioned supervised learning (Brandfonbrener et al., 2022; Paster et al., 2022; Yang et al., 2022), we introduce an additional conditioning variable $g \in \mathcal{G}$ (i.e., user goal). This variable captures overall trajectory-level information, to steer the model toward the goal. Moreover, in implementation, we use observations $\mathbf{o}$ (i.e., visible web page) instead of the actual state $\mathbf{s}$. Our final formulation is expressed as:

$$\mathcal{L}_{\text{LLM}}(\pi) = \mathbb{E}_{\tau\sim\mathcal{D}}[\sum_{t=0}^{L} -\log \pi(\mathbf{a}_t|\tau_{<t}, \mathbf{o}_t, g)].$$

The training of this phase can be skipped or chosen based on the availability and feasibility of acquiring human demonstrations. In our approach to address RQ1 (Effectiveness of Unsupervised Learning), we skip this phase. We limit the human demonstration data to a single category for RQ3 (Unsupervised Domain Adaptation). To investigate RQ2 (Impact of Human Demonstrations), we utilize all the available training data for the supervised training phase.

2.4 Phase Two: Unsupervised Training

The unsupervised learning phase, which forms the core of the proposed agent GLAINTEL, operates without any human demonstrations. This phase is designed to autonomously learn and adapt without relying on expert-guided examples. The objective of the agent is to learn a policy $\pi : \mathcal{O} \times \mathcal{G} \mapsto \mathbb{P}(\mathcal{A})$

that optimizes the expected discounted cumulative rewards for a given goal g. In this work, we leverage PPO algorithm for training, which simultaneously learns a policy $\hat{\pi}$ and a value function $\hat{V} : \mathcal{O} \times \mathcal{G} \mapsto \mathbb{R}$ approximating to the true value $V(\mathbf{s}, g) = \mathbb{E}_{\mathbf{a}\sim\hat{\pi}(\mathcal{O}(\mathbf{s}),g)}\big[\mathcal{R}(\mathbf{s}, \mathbf{a}, g) + \gamma V(\mathcal{T}(\mathbf{s}, \mathbf{a}), g)\big]$. We can calculate the probability of each action $\mathbf{a}_i \in \mathcal{A}$ using the likelihood computed by the model, expressed as: $\hat{\pi}(\mathbf{a}_i|\mathbf{o}, g) = P(\mathbf{a}_i|g)$. That is, the likelihood of choosing each action is calculated based on the probability distributions associated with the tokens that make up the action. This approach ties the action probabilities directly to the distributions of the individual tokens involved in constructing the action. Following (Carta et al., 2023), we incorporate a multilayer perception (MLP) with a single output on top of the last layer of the model to approximate the value V. Specifically, we employ the language modeling head to directly compute the log probabilities of each action $\mathbf{a}_i = \{w_0, \cdots, w_{|\mathbf{a}_i|}\}$ from the dynamic action space given the agent's goal $g \in \mathcal{G}$ and observation $\mathbf{o}_t$ at time t as follows:

$$P(\mathbf{a}_i) = \frac{1}{|a_i|}\sum_{k=0}^{|a_i|} \log P_{\texttt{LM-head}}(w_k|g, \mathbf{o}_t, w_{<k}).$$

Subsequently, employing the softmax operation, we calculate a probability distribution over the action space $\mathcal{A}$ as follows:

$$P(\mathbf{a}_i|g) = \frac{e^{P(\mathbf{a}_i)}}{\sum_{\mathbf{a}_k\in\mathcal{A}} e^{P(\mathbf{a}_k)}}.$$

While the actions comprise multiple tokens, the number of possible actions can vary substantially depending on the current observation (i.e., web page), which introduces additional complexity. This phase is mandatory regardless of whether training is conducted in the optional first phase.

2.5 Phase Three: Inference

In the inference phase, various decoding techniques for action selection can be employed, such as greedy decoding and top-p. Given the well-established nature of these techniques, we omit details and provide key insights only. Greedy decoding, chosen for action selection, has a drawback as it tends to trap the agent in loops, ultimately resulting in suboptimal overall performance. Conversely, opting for top-p sampling can yield a higher success rate, as it provides a theoretical tradeoff between sampling and greedy decoding. However, the process of determining the optimal values for p can be time-intensive. To address these issues, we turn to the Epsilon-Greedy algorithm for action selection during inference. In particular, at a step t, the greedy will choose the action with the highest probability, while the epsilon will sample based on the probability distribution across the action space. This method achieves a higher success rate and an enhanced overall performance, all while avoiding the issue of getting stuck in loops. It is worth noting that a judiciously chosen, small value for epsilon has been employed in our work, eliminating the need for an exhaustive search.

3 Experimental Setup

3.1 WebShop Environment

Webshop (Yao et al., 2022) is a simulated web-based interactive environment with 1.18 million real-world products and 12,087 crowd-sourced text instructions. The goal of the agent is to buy a product with specific attributes and options given natural language instruction. The environment contains 5 different categories, which exhibit significant dissimilarities, particularly in terms of possessing nearly exclusive attributes. For instance, as illustrated in Table 1, a substantial 95.9% of Fashion's attributes are unique to its category.

Human Demonstrations. The Webshop also contains a human demonstration dataset. The human demonstration dataset encompasses a total of 1010 distinct trajectories, distributed across categories. This dataset is created by asking humans to demonstrate how they would query a product and then take different steps in the Webshop environment to buy a product with desired options and attributes.

GLAINTEL has the flexibility to incorporate human demonstrations through optional phase one training. We utilize human demonstration data to quantify the impact of human demonstrations (RQ2) and explore UDA (RQ3). Additionally, GLAINTEL can be trained without any human demonstrations (RQ1).

3.2 Evaluation Methodology

Reward. We assign a reward $r \in [0, 1]$ to the agent after it completes a purchase at the concluding step of an episode. Specifically, the reward is determined by how closely the purchased product matches the specific attributes and options mentioned in the user instructions as follows:

$$r = r_{\text{type}} \cdot \frac{|U_{\text{att}} \cap Y_{\text{att}}| + |U_{\text{opt}} \cap Y_{\text{opt}}| + 1[y_{\text{price}} \leq u_{\text{price}}]}{|U_{\text{att}}| + |U_{\text{opt}}| + 1}$$

Category	# Attributes	% Unique Attributes	# Human Demonstrations
Beauty	143	85.3%	224
Garden	133	87.2%	211
Grocery	117	92.3%	189
Electronics	141	91.4%	169
Fashion	173	95.9%	217

Table 1: Detail about Webshop Environment.

The reward incorporates three main components: U_{att}, U_{opt}, and u_{price}, representing a set of attributes, a set of options, and the price set down in the user's instruction, respectively. Correspondingly, Y_{att}, Y_{opt}, and y_{price} denote the set of attributes, the set of options, and the actual price of the purchased product by the agent. Additionally, r_{type} functions as a text-matching heuristic, assigning a lower reward when the purchased product and the targeted product in the user instruction have similar attributes and options while being different types of products. Interested readers are referred to WebShop (Yao et al., 2022) for details.

Evaluation Metrics. Two evaluation metrics are computed using the rewards obtained from the episodes: (*i*) the Score and (*ii*) the Success Rate. The Score metric represents the average reward across all test episodes multiplied by 100, while the Success rate metric measures the percentage of test episodes in which the full reward (1 out of 1) was attained. Given that our inference step incorporates sampling, the reported Score and Success Rate metrics are averaged by running the model four times. *We provide additional implementation details in Appendix A.*

3.3 Competing Methods

WebShop Baselines (Yao et al., 2022): We consider the following baselines from the WebShop paper: (*i*) rule-based (Rule$_{ws}$), (*ii*) behavioral cloning-based supervised learning (BC$_{ws}$), (*iii*) two reinforcement learning models—one with a transformer text encoder (PG$_{ws}$) and another with an RNN (RNN$_{ws}$), and (*iv*) a hybrid method (BC + PG). Human experts (Human) also set a benchmark for human-level performance.

DRRN (He et al., 2016): DRRN is a classic RL baseline that uses separate neural networks to embed states and actions into embedding vectors. An interaction function (e.g., inner product) then computes the Q-function value for the state-action pair.

Act and ReAct (Yao et al., 2023): The ReAct method is an in-context learning approach using LLMs that combines reasoning and action execution to tackle diverse tasks. In the WebShop environment, ReAct adds reasoning at each step to guide the agent's decisions on exploration, purchasing, and option selection.

WebGUM (Furuta et al., 2024): WebGUM is an instruction-finetuned model, that is further trained on human demonstrations for web navigation.

ASH Prompting (Sridhar et al., 2023): ASH consists of two main components: (*i*) Summarizer condenses observations by retaining only relevant information, and (*ii*) Actor uses this condensed observation to generate the next action.

PIX2ACT (Shaw et al., 2024): PIX2ACT builds upon the Pix2Struct model (Lindenberger et al., 2021), utilizing an image transformer encoder along with a text transformer decoder.

LASER (Ma et al., 2023): LASER is a GPT-4-based method that converts an interactive decision-making task into state space exploration by mapping all possible observations to a finite set of states, with the agent navigating these states through predefined actions specific to each state

Prospector (Kim et al., 2023): The Prospector uses two approaches: the AskAct method, which incorporates self-asking steps in few-shot demonstrations to extract actions from LLMs, and the Trajectory Ranking (TR) method, where LLMs generate diverse trajectories, and the most rewarding one is selected using a reward prediction model.

4 Results

4.1 Quantitative Analysis

RQ1: Effectiveness of Unsupervised Learning. In Table 2, we systematically evaluate the performance of various methods that do not use human demonstrations for training. Starting with RL-based models, our PPO-trained model with 1 million steps (PPO$_{1M}$) emerges as the top performer, achieving a statistically significant score of 72.13 and a success rate of 42.55. Notably, these results surpass those obtained by alternative RL-based approaches, namely PGws, DRRN, and RNN$_{ws}$, underscoring the superior efficacy of the PPO methodology. Among In-context learning methods, the AskAct stands out with the most impressive results. However, even the best-performing AskAct, 70 billion parameters, fails to outperform a smaller model fine-tuned in an unsupervised setting with PPO (PPO$_{1M}$). Specifically, in terms of percentage improvements, the PPO-trained model with 1 million steps (PPO$_{1M}$) outperforms the AskAct by 5.15% on the score metric and approximately

Approach	Name	Model	Parameters	Score	Success Rate
Zero Shot	Random	-	-	33.74	6.80
	Rule$_{ws}$ [1]	-	-	45.60	9.60
	ZSL-Flan-T5	Flan-T5-large	780 Million	41.10	10.30
In-context Learning	Act [2]	PaLM	540 Billion	62.30	30.10
	ASH [4]	CODE-DAVINCI-002	N/A	56.70	30.20
	ReAct [2]	PaLM	540 Billion	66.60	40.00
	AskAct [3]	Llama-2	70 Billion	68.60	42.20
RL-based Method	PG$_{ws}$ [1]	BART, BERT	516 Million	52.50	11.20
	DRRN	GRU	1.2 Million	46.87	11.73
	RNN$_{ws}$ [1]	GRU	5 Million	55.20	17.60
	PPO$_{500K}$ (Ours)	Flan-T5-large	780 Million	68.19	38.55
	PPO$_{1M}$ (Ours)	Flan-T5-large	780 Million	**72.13**	**42.55**
Human	Human [1]	-	-	82.10	59.60

Results are taken from published research: [1] from (Yao et al., 2022), [2] from (Yao et al., 2023), [3] from (Kim et al., 2023), and [4] from (Sridhar et al., 2023).

Table 2: Results from methods in the WebShop environment that do not rely on human demonstration data.

Approach	Name	Model	Parameters	Score	Success Rate
Behavioral Cloning	PIX2ACT [3]	Pix2Struct	282 Million	46.70	NR
	BC$_{ws}$ [1]	BART, BERT	516 Million	59.90	29.10
	BC$_{our}$	Flan-T5-large	780 Million	66.56	37.05
	WebGUM [2]	Flan-T5-XL	3 Billion	67.50	45.00
Hybrid Methods	BC + PG [1]	BART, BERT	516 Million	62.40	28.70
	AskAct + TR (Prospector) [4]	Llama-2, FLAN-T5-XL	70 + 3 Billion	70.20	43.60
	BC + PPO$_{500K}$ (GLAINTEL$_{500K}$)	Flan-T5-large	780 Million	74.60	46.95
	BC + PPO$_{1M}$ (GLAINTEL$_{1M}$)	Flan-T5-large	780 Million	**76.87**	**49.60**

Results are taken from published research: [1] from (Yao et al., 2022), [2] from (Furuta et al., 2024), [3] from (Shaw et al., 2024), and [4] from (Kim et al., 2023).

Table 3: Results from methods in the WebShop environment that use human demonstration data.

0.83% on the success rate metric. This pattern persists when comparing ReAct (540 billion parameters) with PPO$_{1M}$ model. This observation suggests that fine-tuning of small models using RL can yield superior performance compared to in-context learning methods. In addition to RL-based and in-context learning methods, Table 2 includes zero-shot learning methods, including zero-shot Flan-T5 (ZSL-Flan-T5) to quantify the role of unsupervised training.

RQ2: Impact of Human Demonstrations. Table 3 presents the results of various methods incorporating human demonstration. In the behavioral cloning approach, WebGum emerges as the top performer, leveraging the Flan-T5-XL model with 3 billion parameters. It achieves a score of 67.5 and a success rate of 45.0. We also present the results of our fine-tuned Flan-T5-large model (BC$_{our}$) with 780 million parameters. Both models outperform the PIX2ACT and BCws models, which utilize BART and BERT architectures. This notable superiority underscores the effectiveness of instruction-finetuned language models. Turning to hybrid methods, GLAINTEL$_{500K}$, GLAINTEL$_{1M}$, and BC + PG models initially undergo refinement through human demonstrations in a supervised setting, followed by additional fine-tuning in an unsupervised setting using RL. In contrast, Prospector employs the AskAct method (in-context learning) and a reward prediction model, choosing the most rewarding trajectory through supervised learning. Among these approaches, GLAINTEL$_{1M}$ achieves remarkable performance. It attains an exceptional Score of 76.87 and a Success Rate of 49.6. Notably, our approach surpasses all other hybrid and behavioral cloning methods in both metrics.

Effective Utilization of Human Demonstrations: In comparing two variants of the Flan-T5-large model, as presented in Table 3 and Table 2, we focused on one fine-tuned in a supervised setting with human demonstrations (referred to as BC$_{our}$ in Table 3) and another fine-tuned exclusively with PPO for 1 million steps in an unsupervised setting (referred to as PPO$_{1M}$ in Table 2). Surprisingly, the unsupervised model (PPO$_{1M}$) demonstrated an 8.36% higher score and a 14.84% higher success rate compared to the supervised model, which is statistically significant. *This outcome suggests that relying only on human demonstrations does not always lead to superior results.* Moreover, when the supervised model is subjected to further training with PPO, it produces the best results.

Approach	Name	Model	Parameters	Score	Success Rate
RL-based Method	PPO_{1M}	Flan-T5-large	780 Million	72.12	42.55
Hybrid Method	BC + PPO_{1M} (GLAINTEL_{1M})	Flan-T5-large	780 Million	**76.87**	49.6
Unsupervised Domain Adaptation	UDA_{1M}	Flan-T5-large	780 Million	74.69	46.42
State-Space Exploration	LASER(Ma et al., 2023)	GPT-4-0613	N/A	75.6	**50.0**

Table 4: Comparison of the Best Models.

Approach →	Single Domain Behavioral Cloning		Unsupervised Domain Adaptation			
PPO Adaptation Configs →	No PPO (SDBC)		PPO for 500k steps (UDA_{500K})		PPO for 1M steps (UDA_{1M})	
Single-domain Supervision ↓	Score	Success Rate	Score	Success Rate	Score	Success Rate
Fine-tuned on Beauty	64.23	31.41	73.99	45.80	74.49	45.85
Fine-tuned on Garden	64.79	34.76	73.97	44.70	75.27	47.5
Fine-tuned on Grocery	61.80	27.50	73.83	45.75	74.91	47.60
Fine-tuned on Electronics	62.03	30.97	73.46	45.25	74.41	44.5
Fine-tuned on Fashion	62.54	31.60	73.37	44.45	74.36	46.65
Average →	63.07	31.24	73.72	45.19	74.68	46.42

Table 5: The results of unsupervised domain adaptation and single domain methods in the WebShop environment.

Comparison between the Best Models: We present the results from the best models in Table 4. Notably, GLAINTEL_{1M} achieves a state-of-the-art score (i.e., 76.87) surpassing all other models. Surprisingly, our model, based on Flan-T5-Large (780 million parameters), has outperformed the LASER method, which relies on the latest GPT-4 model with extensive handcrafted prompt, in terms of the Score metric. It also achieves comparable performance in terms of Success Rate (49.6 vs 50.0). These findings strongly suggest that a model, when further fine-tuned with PPO after supervised training, can deliver superior results, even with a relatively smaller model size.

RQ3: Unsupervised Domain Adaptation. The Single Domain Behavioral Cloning (SDBC) approach involves fine-tuning a Flan-T5-large model in a supervised setting using demonstrations specific to a particular domain (e.g., Beauty). Subsequently, without any additional refinement for other domains, the model is directly tested using the WebShop environment encompassing all domains. In contrast, UDA takes the Flan-T5-large model fine-tuned in a single domain and further refines it across all domains using PPO in the unsupervised setting. Table 5 presents two versions of UDA: UDA_{500K} and UDA_{1M}. Both UDA methods exhibit superior performance (i.e., statistically significant) in terms of Score and Success Rate metrics when compared to the corresponding metrics of SDBC. This superiority is evident not only on a domain-specific basis but also on the average performance across domains. In particular, concerning the average performance across domains, UDA_{1M} surpasses SDBC by 18.4% in the Score and 48.6% in the Success Rate metrics. This emphasizes the crucial role of unsupervised PPO refinement and its impact on enhancing overall performance.

Role of Supervision in a Single Domain: To compare the UDA results with RL-based ones, we can refer to Table 5 and Table 2, where UDA_{500K} model outperforms the PPO_{500K} in terms of both Score and Success Rate metrics. Similarly, UDA_{1M} surpassed PPO_{1M}. Specifically, the UDA_{1M} model achieves a 3.5% higher Score and a 9.09% higher Success Rate compared to the PPO_{1M} model. Likewise, the UDA_{500K} model attained an 8.1% higher Score and a 17.2% higher Success Rate compared to the PPO_{500K} model. These findings indicate that incorporating single-domain human demonstration supervision significantly enhances the model's capacity for more effective fine-tuning during unsupervised training with PPO. This approach outperforms models that lack any supervised training, which highlights the value of leveraging human demonstrations in the adaptation process.

Learning Curves for PPO training. In Figure 2, the learning curves of Score and Success Rate metrics during PPO fine-tuning are illustrated for various methodologies: the UDA, the hybrid (GLAINTEL) (BC + PPO), and the RL-based PPO. Both the hybrid method and the unsupervised domain adaptation method demonstrate higher sample efficiency compared to the unsupervised method. This aligns with expectations, considering that both the hybrid method and the unsupervised domain adaptation method underwent some level of supervised training before RL fine-tuning – a contrast to the RL-based unsupervised method, which did not.

Approach	Name	Model	Parameters	Score	Success Rate
Hybrid Method	BC + PG	BART, BERT	516 Million	59.25	24
Hybrid Method	BC + PPO$_{1M}$ (GLAINTEL$_{1M}$)	Flan-T5-large	780 Million	78.35	53
State-Space Exploration	LASER	GPT-4-0613	N/A	83.55	56

Table 6: Results of Zero-shot simulation-to-real experiment on eBay.

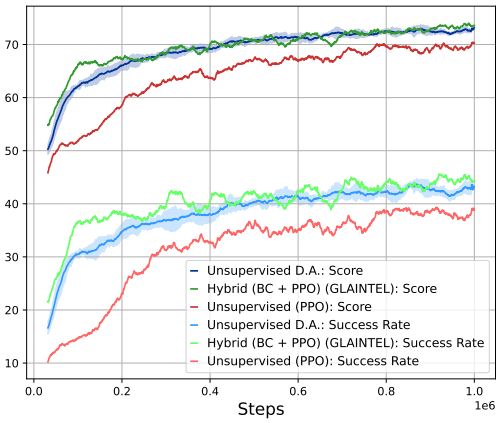

Figure 2: Learning curves of different methodologies: Unsupervised Domain Adaptation (UDA), Hybrid (BC + PPO) (GLAINTEL), and RL-based Unsupervised (PPO).

4.2 Results on Real Website: eBay

We also conduct limited evaluations on a real website: eBay. For this experiment, we evaluate the performance of three methods: (*i*) our best model (GLAINTEL$_{1M}$), (*ii*) the GPT-4-based method LASER, and (*iii*) the WebShop baseline (BC + PG). *It is important to highlight that we used the models trained using the Webshop environment and did not perform any fine-tuning using the eBay website.* Following (Yao et al., 2022), we randomly sampled 100 user instructions to evaluate the performance of these methods. As presented in Table 6, our method GLAINTEL$_{1M}$ significantly outperformed the WebShop baseline (BC + PG) by 32.23% in the Score metric and by 120.83% in the Success Rate metric. Moreover, although LASER, utilizing GPT-4, has slightly higher Score and Success Rate metrics compared to our model GLAINTEL$_{1M}$, we are confident that GLAINTEL$_{1M}$ can achieve comparable or even superior results by enabling of unsupervised training using PPO. Additionally, it is worth noting that our approach utilizes a 780 million parameter model, which is significantly smaller than GPT-4, not to mention the costs associated with GPT-4. *We present an ablation study in Appendix B.*

5 Related Work

Fine-tuning LLMs with RL and Human Feedback. Fine-tuning LLMs with human feedback and reinforcement learning has been studied extensively. (Nakano et al., 2021) developed the WebGPT by fine-tuning the GPT-3 model using behavior cloning and rejection sampling. Moreover, InstructGPT (Ouyang et al., 2022) was developed using the three-step approach: supervised fine-tuning, reward model training, and reinforcement learning via PPO with the help of the trained reward model. Additionally, the authors in (Stiennon et al., 2020) fine-tuned a model that may choose a human-preferred summary, they used this model as a reward function to fine-tune a summarization policy using RL.

Foundation Models for Decision Making. Foundation models possess robust decision-making capabilities, rendering them invaluable across various downstream tasks. For instance, recent works (Ahn et al., 2022; Huang et al., 2022a,b) showcase the application of foundation models in the robotics domain. Moreover, works (Rawles et al., 2023; Wen et al., 2023; Yan et al., 2023; Hong et al., 2023) utilize foundation models to intelligently navigate Android applications. Additionally, the foundation models have been utilized in gaming contexts (, FAIR; Lee et al., 2022; Reed et al., 2022; Fan et al., 2022; Wang et al., 2024; Carta et al., 2023).

Web Navigation. Many benchmarks and datasets exist for the training and assessment of web agents (Yao et al., 2022; Shi et al., 2017; Deng et al., 2024; Zhou et al., 2023; Liu et al., 2018). Researchers have consequently proposed diverse web agents and tested their performance on these benchmarks. The MiniWob++ benchmark is among these benchmarks on which different methods have been applied. For example, (Humphreys et al., 2022) employed a combination of reinforcement learning and behavioral cloning, (Furuta et al., 2024) utilized supervised training on an instruction-fine-tuned LLM, (Liu et al., 2018) introduced Workflow-guided exploration (WGE), and (Gur et al., 2019) trained DQN agents (QWeb network and INET

network). Additionally, the Mind2Web benchmark introduced the MindAct model, synergizing the strength of small and large LLMs (Deng et al., 2024). Additionally, a visual language model named CogAgent was utilized for the benchmark (Hong et al., 2023). (Zeng et al., 2023) presented AgentTuning as another notable approach to tackle the Mind2Web benchmark. Furthermore, considering the Webshop benchmark, various methodologies have been proposed that use in-context learning (Kim et al., 2023; Yao et al., 2023; Sridhar et al., 2023), supervised learning (Furuta et al., 2024; Shaw et al., 2024), and RL (Yao et al., 2022). *Nonetheless, no work has clearly outlined the impact of human demonstrations and the optimal utilization of available demonstration data. Furthermore, UDA remains underexplored.*

6 Conclusion

We introduce GLAINTEL, a flexible agent designed for training across diverse product search scenarios, accommodating situations with limited or no human demonstrations for supervision. We also investigate the optimal utilization of demonstration data, showing that straightforward supervised learning approaches, like behavior cloning, do not yield superior results when using human demonstration data. Through extensive experimental evaluations in the WebShop environment, we highlight the crucial role of the unsupervised training phase employing the PPO algorithm. When combined with supervised learning, this approach achieved results comparable to methods utilizing GPT-4. Additionally, we explore an underexplored scenario where demonstration data is confined to a single domain, we employ UDA techniques to accommodate novel domains. We also present evaluations on a real website, eBay, to showcase the applicability of GLAINTEL in the real world.

Acknowledgments

This work is supported in part by the National Science Foundation (NSF) under grant IIS-2401685.

7 Limitations

In our experiments, we only used the current and previous observations as input to the model. Although including additional observations (e.g., the last four observations) can potentially improve performance, it is important to consider that the increase in the number of observations also expands the size of the context, leading to requirements for higher GPU memory. Moreover, the current architecture relies only on textual descriptions of the environment, without embedding screenshots of web pages or product images. Improving the performance of the agent can be achieved by integrating these visual elements into the model.

It should be noted that other web environments, such as MiniWoB (Shi et al., 2017), have simple, plain backgrounds and minimal interaction within a small area of 160 x 160 pixels. Because of these limitations, we did not assess our method in this environment and considered a more realistic environment, WebShop. However, we plan to evaluate the performance of our approach in other web environments in the future.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Jayant Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego M Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, F. Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. 2022. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

David Brandfonbrener, Alberto Bietti, Jacob Buckman, Romain Laroche, and Joan Bruna. 2022. When does return-conditioned supervised learning work for offline reinforcement learning? In *Advances in Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning.

In *International Conference on Machine Learning*, pages 3676–3713. PMLR.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2024. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. 2017. Openai baselines. https://github.com/openai/baselines.

Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sasha Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyan Shi, Joe Spisak, Alexander Wei, David Wu, Hugh Zhang, and Markus Zijlstra. 2022. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. 2022. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. 2024. Multimodal web navigation with instruction-finetuned foundation models. In *The Twelfth International Conference on Learning Representations*.

Izzeddin Gur, Ulrich Rueckert, Aleksandra Faust, and Dilek Hakkani-Tur. 2019. Learning to navigate the web. In *International Conference on Learning Representations*.

Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. 2016. Deep reinforcement learning with a natural language action space. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1621–1630, Berlin, Germany. Association for Computational Linguistics.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. 2023. Cogagent: A visual language model for gui agents. *Preprint*, arXiv:2312.08914.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9118–9147. PMLR.

Wenlong Huang, F. Xia, Ted Xiao, Harris Chan, Jacky Liang, Peter R. Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. 2022b. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*.

Peter C. Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Alex Goldin, Adam Santoro, and Timothy P. Lillicrap. 2022. A data-driven approach for learning to control computers. In *International Conference on Machine Learning*.

Byoungjip Kim, Youngsoo Jang, Lajanugen Logeswaran, Geon-Hyeong Kim, Yu Jin Kim, Honglak Lee, and Moontae Lee. 2023. Prospector: Improving LLM agents with self-asking and trajectory ranking. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Kuang-Huei Lee, Ofir Nachum, Mengjiao (Sherry) Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, and Igor Mordatch. 2022. Multi-game decision transformers. In *Advances in Neural Information Processing Systems*, volume 35, pages 27921–27936. Curran Associates, Inc.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE.

Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. 2021. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5987–5997.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. 2018. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations (ICLR)*.

Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, and Dong Yu. 2023. LASER: LLM agent with state-space exploration for web navigation. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Kyle Mahowald, Anna A Ivanova, Idan A Blank, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. 2023. Dissociating language and thought in large language models: a cognitive perspective. *arXiv preprint arXiv:2301.06627*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Keiran Paster, Sheila McIlraith, and Jimmy Ba. 2022. You can't count on luck: Why decision transformers and rvs fail in stochastic environments. *Advances in neural information processing systems*, 35:38966–38979.

Dean A. Pomerleau. 1988. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley D. Edwards, Nicolas Manfred Otto Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. 2022. A generalist agent. *Transactions on Machine Learning Research*, 2022.

Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina N Toutanova. 2024. From pixels to ui actions: Learning to follow instructions via graphical user interfaces. *Advances in Neural Information Processing Systems*, 36.

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. 2017. World of bits: An open-domain platform for web-based agents. In *International Conference on Machine Learning*, pages 3135–3144. PMLR.

Abishek Sridhar, Robert Lo, Frank F. Xu, Hao Zhu, and Shuyan Zhou. 2023. Hierarchical prompting assists large language model on web navigation. In *Conference on Empirical Methods in Natural Language Processing*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*.

Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. 2023. Empowering llm to use smartphone for intelligent task automation. *ArXiv*, abs/2308.15272.

An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, Zicheng Liu, and Lijuan Wang. 2023. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. *ArXiv*, abs/2311.07562.

Mengjiao Yang, Dale Schuurmans, Pieter Abbeel, and Ofir Nachum. 2022. Dichotomy of control: Separating what you can control from what you cannot. *International Conference on Learning Representations*.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757. Curran Associates, Inc.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2023. Agenttuning: Enabling generalized agent abilities for llms. *Preprint*, arXiv:2310.12823.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Hyperparameter	Value
Number of Epochs	10
Learning Rate	2×10^{-5}
Warmup Steps	100
Weight Decay	0.01
Batch Size	32
Adam Optimizer Epsilon	10^{-8}
Adam Optimizer β_1	0.9
Adam Optimizer β_2	0.999

Table 7: Supervised Learning Hyperparameters.

Hyperparameter	Value
# of collected transitions between two updates	640 (16 × 40)
Number of epochs per update	1
Batch Size	8
Learning Rate	10^{-6}
Adam Optimizer Epsilon	10^{-5}
Adam Optimizer β_1	0.9
Adam Optimizer β_2	0.999
Discount Factor	0.99
Lambda for Generalized Advantage Estimate	0.99
Entropy Loss Coefficient	0.01
Value Loss Coefficient	0.5
Maximum Gradient Norm	0.5
Clipping Epsilon	0.2

Table 8: Unsupervised Learning Hyperparameters.

A Implementation Details

Our implementation operates on a client-server architecture, with the training scripts serving as the client and communicating requests to LLM servers. Specifically, a master server manages these requests, distributing them across multiple LLM servers. Once each LLM server completes its computations, the master server consolidates the results and sends them back to the training script. Furthermore, we use vertical model parallelism, enabling the parallelization of individual LLMs across multiple GPUs. In our experiments, we utilized a single LLM, Flan-T5-Large, with 780 million parameters. This model was parallelized across 4 Nvidia V100 32GB GPUs. We incorporated the last two observations as the model input and an encoder context size of 1024.

To train the agent using the human demonstrations, we used the Trainer library provided by Huggingface [2]. We employed the Adam optimizer, and for the remaining hyperparameter values, refer to Table 7. In our unsupervised learning phase, we leverage the PPO algorithm, and the complete values of hyperparameters can be found in Table 8.

Configs →	SL (one cat) + PPO (500k)		PPO (500k)	
Model ↓	Score	Success Rate	Score	Success Rate
Flan-T5	**73.72**	**45.19**	**68.18**	**38.55**
T5	71.85	43.10	52.07	25.35

Table 9: Ablation Study (T5 vs Flan-T5)

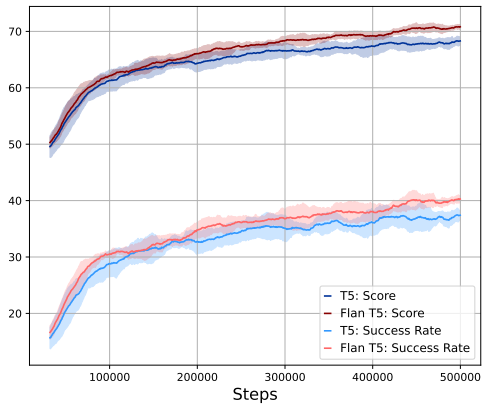

Figure 3: Hybrid setting: BC + PPO: Flan-T5 is more sample efficient than T5 model.

B Ablation Study

Flan-T5 vs T5. We employed two models of identical size, each with 780 million parameters: Flan-T5-Large and T5-Large. The results, as presented in Table 9, demonstrate that adopting the Flan-T5-Large model instead of T5-Large leads to a substantial improvement of 30.93% in the Score and a remarkable 52.07% increase in the Success Rate in the unsupervised setting (PPO). Furthermore, in the domain adaptation scenario, we observed a 2.60% Score enhancement and a 4.85% improvement in the Success Rate. Moreover, Figure 3 demonstrates that employing the Flan-T5 model over the T5 model results in better sample efficiency. Specifically, both Score and Success Rate metrics exhibit faster growth during PPO fine-tuning in the Flan-T5 model compared to the T5 model. This outcome was anticipated as the Flan-T5 model enjoys the advantage of being fine-tuned on user instructions, a benefit not shared by the T5 model.

[2]Trainer: https://huggingface.co/docs/transformers/main_classes/trainer

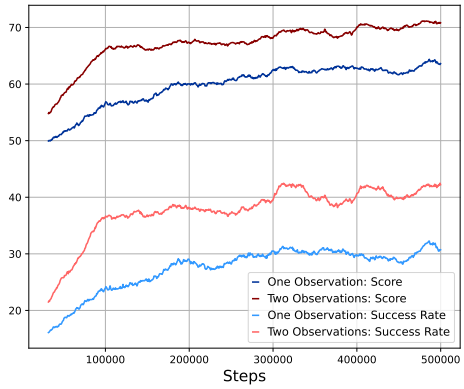

Figure 4: The model is more sample efficient when we feed it with the last two observations.

Configs →	SL (all cats)		SL + PPO (500k)	
	Score	Success Rate	Score	Success Rate
2 observations	**66.55**	**37.05**	**74.60**	**46.95**
1 observation	60.20	27.20	65.29	33.60

Table 10: Ablation Study (2 observations vs 1 observation)

2 Observations vs 1 Observation. As demonstrated in Table 10, combining the present observation state with the preceding observation state to create a historical context and subsequently providing the model with this new observation containing both leads to a notable 10.54% boost in the Score and a remarkable 36.21% improvement in Success Rate in the supervised setting. This substantial enhancement is equally observable in the context of the hybrid method (SL + PPO) where the supervised training is coupled with unsupervised training (PPO), resulting in a significant 14.26% increase in the Score and an impressive 39.73% improvement in Success Rate. Additionally, during the training, we noticed that employing a historical context (having the current and last observations) as input enhances the sample efficiency for the agent compared to using just one observation (see Figure 4). Specifically, Score and Success Rate metrics show a swifter increase with fewer steps when leveraging two observations (historical context) as input, while the progression is notably slower when utilizing only a single (or current) observation.

Comparison of Decoding Methods. In Table 11, we compare the performance of four different decoding methods: (*i*) Epsioln-Greedy algorithm (with epsilon value of 0.2), (*ii*) Sampling with top_p (with top_p = 0.8 and top_k = 0.0),(*iii*) Sampling with no top_p and no top_k, and (*iv*) Argmax. These results are determined by averaging the results achieved from models trained with different techniques and settings, including RL and UDA, among others. These results show that, on average, the Epsilon-Greedy algorithm consistently attains the best results during inference, with a Score of 68.23 and a Success Rate of 39.29. Following closely, the nucleus sampling (top_p) method has lower Scores and Success Rates of 66.25 and 37.32, respectively. In the third position, traditional sampling produces a score of 65.92 and a Success Rate of 36.41. The worst outcomes are associated with the Argmax method, primarily since Argmax frequently causes the web agent to become stuck in a loop. In simpler terms, the web agent ends up repeatedly navigating back and forth between web pages.

Comparison	Score	Success Rate
Epsilon-Greedy algorithm	**68.23**	**39.29**
Sampling with top_p	66.25	37.32
Sampling	65.92	36.41
Argmax	57.92	35.59

Table 11: Ablation Study (Decoding Methods)