Identifying User Goals from UI Trajectories

Anonymous ACL submission

Abstract

Identifying underlying user goals and intents has been recognized as valuable in various settings, such as personalized agents, improved search responses, advertising, user analytics and more. In this paper we propose leveraging an additional signal for identifying user intents, namely by observing users' interactions within UI environments. To that end, we introduce the task of goal identification from observed UI trajectories, aiming to infer the user's intended task based on their UI interactions. We propose a novel evaluation metric to assess whether two task descriptions are paraphrases within a specific UI environment. By Leveraging the inverse relation with the UI automation task, we utilized Android and web datasets for our experiments. Using our metric and these datasets, we conducted experiments comparing the performance of humans and state-of-the-art models, specifically GPT-4 and Gemini-1.5 Pro. Our results demonstrate that both Gemini and GPT underperform compared to humans, highlighting significant room for improvement.

1 Introduction

Autonomous agents that interact with GUIs to complete tasks for users have drawn increasing interest (Hong et al., 2023; Gur et al., 2023; Yang et al., 2023b). These agents interpret user-provided instructions and iteratively interact with GUIs to complete the desired task. In this work, we propose empowering agents with the ability to identify the underlying goals of users from their observed activity within the GUI environment. Such understanding has the potential to significantly increase agents' utility for users, while providing more personalized and effective assistance (Li et al., 2024).

Consider the scenario in Figure 1 where a user books flight tickets for a vacation. An ideal agent would observe these actions, understand the underlying user goal and then proactively suggest booking a hotel for the same dates, and make the dates visible in the calendar.

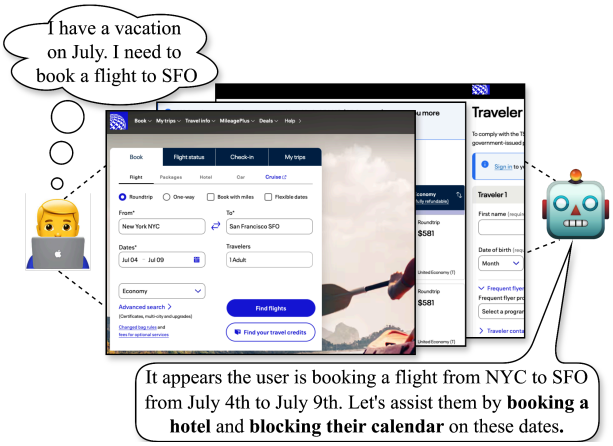

Figure 1: An example of a user performing a flight booking task. The agent first observes the UI interactions, comprehends the task's essence, and then offers help with related tasks, like booking a hotel and blocking calendar dates. We focus on the first part, comprehending the task by observing the UI interactions.

Our work extends a long line of research on recognizing user goals from their observed behaviour, including intent, activity and plan recognition. However, most prior work addressed settings in which the input consists of natural language user utterances, like search queries, dialog utterances or social media posts, while the task was perceived as structured classification, selecting a category label from a predefined list. Our work, to the best of our knowledge, is the first to identify user goals from UI interactions, while providing a natural language description of the user goal as output. We refer to Appendix A for an extensive overview.

In this paper, we first define the task of generating a natural language description of underlying user goals from observed UI trajectories, that is, from multi-modal (text and screen image) traces of user-system interactions. A challenging aspect of the task is its inherent ambiguity, since multiple

user goals can often lead to the same UI activity. Second, we observe that our task can be framed as the inverse of the known *UI automation task*, where an agent needs to perform a sequence of UI actions given a natural language description of the user's goal (Li et al., 2023; Wen et al., 2023). We further introduce manual and automatic evaluation protocols, assessing whether the predicted and gold task descriptions are paraphrases within the given UI context. Subsequently, we conducted experiments over both web and Android UI sessions, leveraging existing UI automation datasets while swapping the input and output roles. Over these data, we compared and analyzed the performance of humans and state-of-the-art multi-modal models, showing that there is substantial room for modeling improvements in future work.

Overall, we offer the following contributions: (1) introducing and formalizing the task of goal identification from UI trajectories; (2) suggesting how existing datasets for UI automation can be leveraged for our task, viewing one task as the "reversed" of the other; (3) introducing manual and automatic evaluation methodologies; (4) evaluating both humans and model performances. We propose that these contributions would trigger research on this timely challenge.

2 Task Definition

Given an observed UI trajectory (see below) performed by a user with the intention to complete a certain task, our goal is to recover the user's original intent from the observable trajectory. As mentioned in § 1, this setting is effectively the inverse problem of the known *UI Automation* task. We therefore adopt their input and output definitions, swapping their roles, which enables the use of UI automation datasets for our task as well.

Accordingly, our input is a UI trajectory — a sequence of individual interaction steps between the user and the system along the session. Each step consists of a snapshot of the UI content at that moment, along with the corresponding action the user took at that step (see Appendix B for the specific trajectory formats). From this trajectory, our goal is to generate a natural language description that accurately captures the user's intended task. Within the scope of this paper, we address the core setting of the intent identification task and therefore assume that the observed UI trajectory indeed successfully fulfills the underlying user intent.

The intent identification task, similar to other text generation tasks like summarization where multiple valid outputs can exist, is inherently ambiguous, mostly because the same trajectory may fulfill multiple intents, often due to varying specificity levels of the original user intent (as captured in the dataset). For example, when observing a Sushi restaurant booking, the user might have asked for that specific Sushi place, or more broadly for *some* restaurant in that area, or of that cuisine type. When a model identifies an intent from the given trajectory, we expect it to predict the most likely one, as reflected in the dataset distribution.

3 Evaluation Methodology

This section outlines our evaluation methodology. Given an input UI trajectory and a corresponding gold task description, we assess whether a *predicted* task description matches the gold reference. We start with necessary definitions (Section 3.1), followed by a human and automatic evaluation protocols (Sections 3.2 and 3.3).

3.1 Definitions

Task Fulfillment by a Trajectory As mentioned in § 2, we assume an observed UI trajectory fulfills the underlying user's intended tasks. Therefore, a predicted task description not fulfilled by the input trajectory is erroneous and does not match the gold task description.

Inspired by the taxonomy proposed by Zhou et al. (2023), we differentiate between fulfillment for information-seeking intents and transactional intents (e.g. purchasing an item or changing settings). For the latter, fulfillment is achieved upon successfully completing the specific requirement outlined in the task. For information-seeking intents, fulfillment is achieved when the trajectory provides the necessary information sought in the user intent. We note that a fulfilling trajectory may provide some extra information beyond the intent, if such additional information is inherently bundled in the UI environment together with the sought information.

Satisfaction Relation between Tasks We next aim to specify a matching criterion between two task descriptions, specifically a predicted one and the corresponding gold-reference. Given two task descriptions A and B and a UI environment, we say that A *satisfies* B in that environment if every reasonable trajectory that fulfills A would also fulfill B. In essence, this means that completing task A

necessarily results in completing task B, making B a more general task than A in that UI environment. For instance, the task "Purchase the earliest train ticket to Edinburgh" satisfies the task "Purchase a train ticket to Edinburgh" but not vice versa.

Building on these definitions, we consider a predicted task description to successfully *match* the gold description if the two mutually satisfy each other, and *partial match* when only one satisfies the other. Essentially, matching tasks can be considered as paraphrases of the same intent within the context of the UI environment. For instance, the tasks "Find a large dining table" and "Find a dining table for 10-12 people" match each other if the UI environment considers 10-12 people as large. We highlight the relation between *match* and ambiguous trajectory, which fulfills multiple task descriptions that do not match each other.

3.2 Human Evaluation Protocol

As with many text generation tasks, human evaluation is essential due to the limited reliability of automatic evaluation metrics. In our case, the annotator observes the gold and predicted task descriptions, with the corresponding trajectory, and assesses (1) whether the trajectory fulfills the predicted task, and (2) whether each of the predicted and gold task descriptions satisfies the other. Additionally, annotators validated the given data instance, excluding from the evaluation noisy instances in the original datasets (details in Appendix C).

To assess the quality of our proposed metric, we randomly sampled 50 instances from each of our datasets (see § 4.1) and generated task descriptions using our two baseline models (see § 4.2). We then measured the pairwise inter-annotator agreement among three of the authors, resulting in an average Cohen's Kappa of 0.79 and 0.77 for fulfillment and satisfaction judgments, respectively, in the web dataset, and 0.91 and 0.86 in the Android dataset. These agreement levels are considered high according to Kappa values, which justified our decision to manually evaluate each baseline model in our experiments using a single annotator.

3.3 Automatic Evaluation Metric

We propose utilizing a Large Multimodal Model (LMM) as an automatic evaluator for the satisfaction criteria. Recent advancements in LMMs, such as those demonstrated in (He et al., 2024) and (Pan et al., 2024), have shown promising results in employing GPT-4 (Achiam et al., 2023) to assess task completion by autonomous agents. Building on this, we leverage the latest GPT-4o model as the automatic evaluator, to determine whether two task descriptions are mutually satisfied in the context of the trajectory (prompt details are in Appendix E). Measuring agreement with our human evaluation yielded a Kappa value of 0.48 (moderate agreement), suggesting a potential utility of model-based automatic evaluation for development cycles while highlighting the need for manual evaluation.

4 Datasets and Baseline Models

4.1 Evaluated Datasets

Given that our task is the inverse of UI automation task, leveraging the datasets created for UI automation is a natural choice. In these datasets, humans interact with a UI to complete a given task.

For our experiments, we explore two UI environments: web and Android. We utilize the Mind2Web dataset (Deng et al., 2023) for the web environment, as it is the most widely used benchmark for autonomous web agents. The dataset was created by curating diverse tasks across popular websites, with annotators performing a series of actions to complete the goal. For Android, we used the prominent Android in the Wild (AitW) (Rawles et al., 2023) dataset, while focusing on its quality-filtered subset Android in the Zoo (AitZ) (Zhang et al., 2024) (details in Appendix B).

4.2 Models

Given the multimodal nature of UI trajectories, our models must be adept at interpreting both text and images. We selected two state-of-the-art LMMs, Gemini 1.5 Pro (Reid et al., 2024) and GPT-4. These models are at the forefront of handling combined text and image inputs and offer a sufficiently large context window for our experiments.

In our experiments, the model was guided through a Chain-of-Thought (Wei et al., 2022) pro-

Dataset		Non Match	Partial Match	Match (↑)
Mind2Web	H vs. G	0.07	0.13	0.80
	H_1 vs. H_2	0.09	0.10	0.81
AitZ	H vs. G	0.02	0.22	0.76
	H_1 vs. H_2	0.02	0.04	0.94

Table 1: Manual evaluation of human generated task descriptions, at the different levels of match criteria (determined by the satisfaction relation) between annotators (H_i) and gold task descriptions (G), as well as among annotators. (H) represents average scores.

		Manual Eval				Automatic Eval		
Dataset	Model	Fulfillment	Non Match	Partial Match	Match (↑)	Non Match	Partial Match	Match (↑)
Mind2Web	GPT-4	0.86	0.14	0.42	0.44	0.18	0.40	0.42
	Gemini-1.5	0.87	0.20	0.22	**0.58**	0.18	0.28	**0.54**
AitZ	GPT-4	0.78	0.22	0.19	**0.59**	0.32	0.34	0.34
	Gemini-1.5	0.88	0.17	0.26	0.57	0.37	0.27	**0.36**

Table 2: Manual and automatic evaluation scores of fulfillment relation and the different levels of match criteria (determined by the satisfaction relation), between model predictions and gold task descriptions.

cess, to analyze the trajectory in a step-by-step manner (prompt in Appendix E).

In web, inspired by SeeAct (Zheng et al., 2024) we drew a red bounding box around the element that the user interacted with to guide the model's attention, as well as to break ambiguity where textual action descriptions were not sufficient. To further focus models, we also truncated lengthy web images based on the bounding box position. (details in Appendix C). For Android, no special modifications needed, actions are overlaid on the screenshots, and the screenshots are naturally smaller.

5 Experiments

5.1 Human Performance Evaluation

To asses task difficulty and establish a baseline for models, two NLP practitioners, unfamiliar with the datasets, independently composed 50 task descriptions from Mind2Web, and 50 from AitZ. An annotator then evaluated[1] them as explained in § 3.2.

The results, summarized in Table 1, reveal more matches between the human annotators and the gold task descriptions in Mind2Web compared to AitZ. Upon analysis of the disagreements, we found that the gold descriptions in Android were often more specific than those provided by human annotators. In most cases, the gold tasks were already fulfilled by the trajectory, resulting in no clear interactions that indicate the user goal. For example, if the task was "Turn WiFi on" and the WiFi was already on, the annotators inferred a more general task, such as "Show WiFi settings". Further analysis of non-matching records across both datasets revealed that ambiguous trajectories caused human disagreements. Detailed analysis in Appendix D.

5.2 Model Performance Evaluation

Model evaluation included manual assessment of 100 predictions from each dataset and automatic evaluation using the entire test sets: 1,013 from Mind2Web and 506 from AitZ (see Appendix C).

The results, outlined in Table 2, reveal that Gemini outperforms GPT on the Mind2Web dataset, achieving higher match scores but still falling short of humans. In AitZ, both models perform comparably. Gemini tends to be more specific and detailed than the gold references, while GPT often generates more general, abstract goals. Both models, in certain instances, misidentified the actual user intent (e.g. "Watch top rated movie trailer" vs. "Watch The Dark Knight trailer") and exhibit limitations in visual screen understanding, leading to missing details, incorporating irrelevant information and hallucinations of non-existent information. Detailed discussion in Appendix D.

These results underscore the complexity of accurately capturing user intent. Our experiments also concluded few-shot learning. However, in our setting, this is challenging due to multiple images, textual inputs and the thought process within example. Adding a single example deteriorated results, likely because the large context size and the difference between the exemplar and the test trajectory.

6 Conclusion and Future Work

In this paper, we introduced the novel task of identifying user goals from UI trajectories and proposed a reliable evaluation methodology. Our experiments on Android and web datasets reveal a significant gap between humans and state-of-the-art multi-modal models, which is analyzed in detail.

Future work may include fine-tuning experiments and enhancing visual understanding for UI environments. We propose testing our task by evaluating its utility on downstream tasks like agent personalization and suggestions. Additionally, expanding the scope to other GUIs like iOS and Windows would broaden the impact of our findings.

[1]During the evaluation it was observed that human tasks were inherently fulfilled by the trajectory.

Limitations

Our study is subject to several limitations due to the nature of the datasets used. First, both datasets primarily include English-language websites accessed in the U.S., thereby limiting the study to English-language interactions only. Second, in real-world scenarios, we believe that (1) user trajectories may be interleaved between multiple tasks as users adjust their objectives in real time or are interrupted by other tasks, (2) users might have more ambiguous goals that evolve during their interaction with the user interface and (3) users might be less proficient with computers or phones, leading to noisier trajectories that are more challenging to identify and interpret intent from. Moreover, in the Mind2Web dataset, all tasks were limited to interactions within the same website, not encompassing multi-website tasks. This constraint may not fully represent the complexity of real-world web usage.

Lastly, this study focuses solely on Android and web environments. These environments might exhibit different task distributions compared to other user interface environments such as iOS and Windows, potentially limiting our findings for Android and web environments.

Ethical Considerations

The development of autonomous agents, while holding great potential for innovation, raises important ethical considerations. Our research focuses on understanding user intent from recorded UI trajectories, and it is crucial to acknowledge the potential privacy implications of tracking user activity. Ensuring the security and protection of this sensitive data is of the utmost importance. Employing techniques like on-device processing, anonymization, or other privacy-preserving methods can help mitigate risks and ensure user data remains protected. It is essential for researchers and developers to proactively address these concerns to foster trust and responsible innovation in the field of autonomous agents.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Eugene Charniak and Robert P Goldman. 1993. A bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*.

Robert P Goldman, Christopher W Geib, and Christopher A Miller. 2013. A new model of plan recognition. *arXiv preprint arXiv:1301.6700*.

Victor Machado Gonzaga, Nils Murrugarra-Llerena, and Ricardo Marcacini. 2021. Multimodal intent classification with incomplete modalities using text embedding propagation. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, pages 217–220.

Roger L Granada, Ramon Fraga Pereira, Juarez Monteiro, Duncan Dubugras Alcoba Ruiz, Rodrigo Coelho Barros, and Felipe Rech Meneguzzi. 2017. Hybrid activity and plan recognition for video streams. In *Proceedings of the 31st. AAAI Conference: Plan, Activity and Intent Recognition Workshop, 2017, Estados Unidos.*

Vineet Gupta, Devesh Varshney, Harsh Jhamtani, Deepam Kedia, and Shweta Karwa. 2014. Identifying purchase intent from social posts. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8, pages 180–186.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2023. A real-world webagent with planning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*.

Rejwanul Haque, Arvind Ramadurai, Mohammed Hasanuzzaman, and Andy Way. 2019. Mining purchase intent in twitter. *Computación y Sistemas*, 23(3):871–881.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. Webvoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*.

Jun Hong. 2001. Goal recognition through goal graph analysis. *Journal of Artificial Intelligence Research*, 15:1–30.

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

Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. 2009. Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*, pages 471–480.

Momal Ijaz, Renato Diaz, and Chen Chen. 2022. Multimodal transformer for nursing activity recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2065–2074.

Henry A Kautz et al. 1991. A formal theory of plan recognition and its implementation. *Reasoning about plans*, pages 69–125.

Imran Ullah Khan, Sitara Afzal, and Jong Weon Lee. 2022. Human activity recognition via hybrid deep learning based model. *Sensors*, 22(1):323.

Sahil Kuchlous and Madhura Kadaba. 2020. Short text intent classification for conversational agents. In *2020 IEEE 17th India Council International Conference (INDICON)*, pages 1–4. IEEE.

Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. 2011. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82.

Stefan Larson and Kevin Leach. 2022. A survey of intent classification and slot-filling datasets for task-oriented dialog. *arXiv preprint arXiv:2207.13211*.

Wei Li, Fu-Lin Hsu, Will Bishop, Folawiyo Campbell-Ajala, Oriana Riva, and Max Lin. 2023. Uinav: A maker of ui automation agents. *arXiv preprint arXiv:2312.10170*.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. 2020. Mapping natural language instructions to mobile ui action sequences. *arXiv preprint arXiv:2005.03776*.

Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, et al. 2024. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*.

Lin Liao, Dieter Fox, and Henry Kautz. 2005. Location-based activity recognition. *Advances in neural information processing systems*, 18.

Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. 2024. Autonomous evaluation and refinement of digital agents. *arXiv preprint arXiv:2404.06474*.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Android in the wild: A large-scale dataset for android device control. *arXiv preprint arXiv:2307.10088*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Jetze Schuurmans and Flavius Frasincar. 2019. Intent classification for dialogue utterances. *IEEE Intelligent Systems*, 35(1):82–88.

Gita Sukthankar, Christopher Geib, Hung Hai Bui, David Pynadath, and Robert P Goldman. 2014. *Plan, activity, and intent recognition: Theory and practice*. Newnes.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Hao Wen, Hongming Wang, Jiaxuan Liu, and Yuanchun Li. 2023. Droidbot-gpt: Gpt-powered ui automation for android. *arXiv preprint arXiv:2304.07061*.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023a. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2023b. Appagent: Multimodal agents as smartphone users. *arXiv preprint arXiv:2312.13771*.

Hanlei Zhang, Hua Xu, Xin Wang, Qianrui Zhou, Shaojie Zhao, and Jiayan Teng. 2022. Mintrec: A new dataset for multimodal intent recognition. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1688–1697.

Jiwen Zhang, Jihao Wu, Yihua Teng, Minghui Liao, Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang. 2024. Android in the zoo: Chain-of-action-thought for gui agents. *arXiv preprint arXiv:2403.02713*.

Lu Zhang, Jialie Shen, Jian Zhang, Jingsong Xu, Zhibin Li, Yazhou Yao, and Litao Yu. 2021. Multimodal marketing intent analysis for effective targeted advertising. *IEEE Transactions on Multimedia*, 24:1830–1843.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*.

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*.

A Related Work

In this appendix, we provide an overview of key related research areas, specifically intent recognition, activity recognition, and plan recognition.

Intent Recognition Intent recognition, also referred to as intent classification, is a well-established field focused on identifying user intentions based on specific inputs. Traditionally, most research in this domain has concentrated on textual inputs, such as individual messages or utterances, which are then categorized into predefined intent classes. For instance, studies like (Schuurmans and Frasincar, 2019; Kuchlous and Kadaba, 2020; Larson and Leach, 2022) have worked on improving intent classification within dialogue systems across various domains, focusing on the classification of short texts or single utterances into predetermined intent categories (e.g. "find a train from Barcelona to Madrid" would be classified into a system intent called "find_train"). Others, classified social media posts to determine whether they express an intention to make a purchase (Gupta et al., 2014; Haque et al., 2019). To enhance understanding of user queries and infer its underlying intent, some works leverage external knowledge sources to get better results (Hu et al., 2009).

In addition to text-based inputs, multimodal methods have been developed to incorporate various types of inputs. For example, (Zhang et al., 2021) and (Gonzaga et al., 2021) investigate the use of both images and text: the former analyzes social news content to identify marketing intents and to classify intent topics, while the latter combines image and textual data to classify social media posts, aiming to identify the writers' intent such as provocative, informative, promotive and more. Zhang et al. (2022) introduces a multimodal dataset for intent recognition in TV series, where inputs include visual, auditory, and textual data, while the outputs are classified into one of 25 possible intent categories.

Activity Recognition Activity recognition, as defined by Sukthankar et al. (2014) and closely related to intent recognition, involves identifying specific human activities based on a series of observations, often through sensor data. While intent recognition focuses on understanding user intentions, activity recognition mostly concerned with classifying physical actions. Typically, systems in this domain are developed to classify these activities using sensory inputs. Khan et al. (2022) uses a neural network trained on 2D skeletal data captured by a motion sensor to classify human poses; Ijaz et al. (2022) integrated accelerometer signals along with skeletal data to recognize and categorize nurse activities into 12 distinct types. A prominent study (Kwapisz et al., 2011) developed a supervised learning algorithm that uses accelerometer signal from Android smartphones to classify physical activities like walking, jogging, sitting, and standing. Moreover, research such as (Liao et al., 2005) utilizes GPS traces to classify human activity and label significant locations, employing relational Markov networks to achieve this.

Plan Recognition In a plan recognition problem, a system is given a series of actions performed by an agent and is expected to infer the overall plan or goal which explains those actions (Kautz et al., 1991). The distinction between activity recognition and plan recognition is the difference between recognizing a single activity and recognizing the relationships between a set of such activities that result in a complete plan or goal. In this field, foundational works like (Charniak and Goldman, 1993; Goldman et al., 2013) rely on structured inputs and a plan library —– a collection of possible plans and their associated goals — to construct probabilistic models that infer the plan from observed actions. Alternative approaches, such as in (Hong, 2001), develop algorithms that do not depend on plan libraries; instead, they utilizing in-domain knowledge and applying this to UNIX system commands to understand broader user goal. A hybrid approach is demonstrated (Granada et al., 2017), where the authors combine CNN models to detect human activities from a video stream of meal preparation and apply symbolic reasoning with a plan library to recognize the overall plan (e.g., actions like "breaking eggs" and "mixing ingredients" might be recognized as part of a plan to create an omelet).

B Datasets Overview

Web We aim to give a brief overview of how Mind2Web was collected, and its format in more depth. The data collection process involved selecting numerous popular websites across five top-level domains. Annotators, guided by seed tasks from GPT, proposed diverse tasks for each website. They demonstrated and recorded how to complete these tasks, resulting in a trajectory of actions and screenshots. Each action is defined by a pair consisting of a Target Element and an Operation. The Target Element is an interactable element on the current web page, such as buttons, input fields, or drop down menus. The Operation refers to the specific

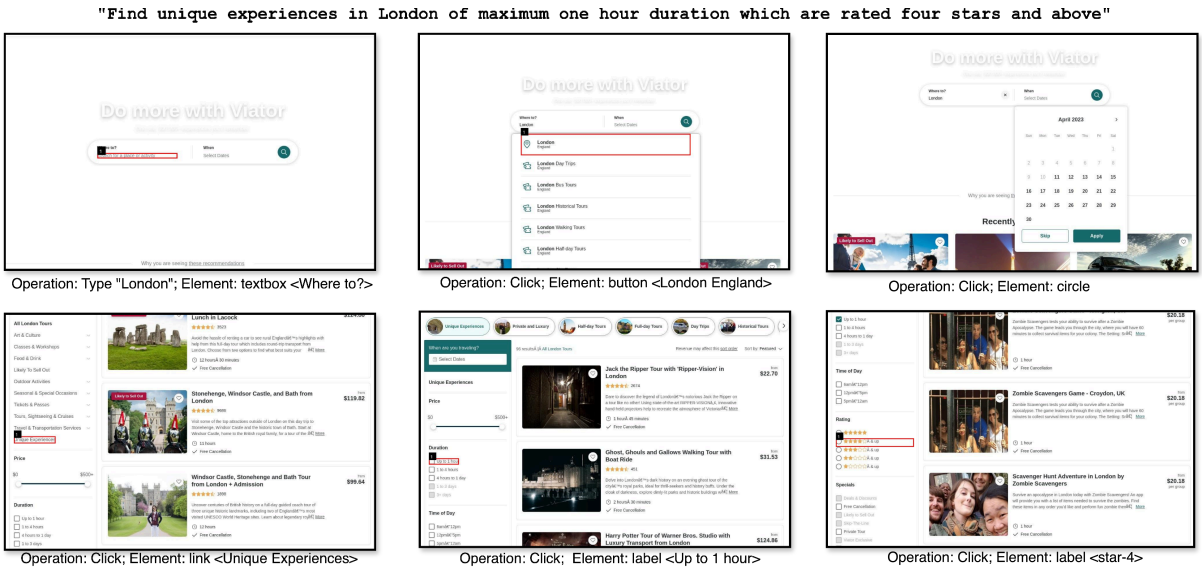

Figure 2: An instance from Mind2Web, representing a full trajectory accomplishing the task description above.

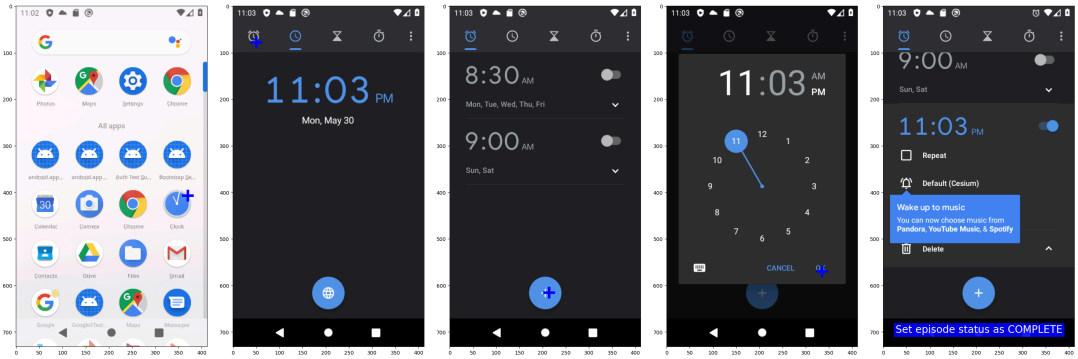

Figure 3: An instance from AitW, representing a full trajectory accomplishing the task "Set an alarm". The blue plus sign indicates the area on the screen where the tap occured.

action to be executed on the Target Element, with Mind2Web supporting three primary operations: Click (including actions like clicking, hovering, and pressing Enter), Type (which involves entering text into input fields and requires an additional value for the text to be typed), and Select Option (which involves selecting an option from a drop down menu or similar element and also requires an additional value for the option to be selected). Notably, these actions were automatically produced during the time users were recorded completing the task, eliminating the need for additional manual labor. Figure 2 demonstrates a single instance of the data, where textual action descriptions are presented below the corresponding screenshot.

Android The Android in the Wild (AitW) dataset stands out due to its extensive variety of tasks, covering 4 domains: Google-Apps, Install, Web-Shopping, and General, as well as a single-step domain excluded from this paper's analysis. The dataset consists of a substantial collection of high-level instructions, trajectories of varying lengths, and a notable variety of apps and websites. The Android in the Zoo (AitZ) dataset was sampled from AitW to reduce tasks redundancy and to filter erroneous data points, resulting in more unique and higher-quality task descriptions.

The high-level goal instructions in AitW were sourced from various sources, including humans (both crowd-sourced and the authors themselves), LLM-generated prompts, and technical documentation such as PixelHelp (Li et al., 2020). The creation process involved human annotators performing tasks on Android emulators, with their gestures being recorded.

The episodes were recorded on mobile devices running four different versions of Android. Each episode contains natural language instructions and

observation-action pairs. The observations are screenshots, while the actions are one of three types: tap, drag, or typing. Gesture actions are represented as taps and drags at specific <x,y> coordinates on the screen.

In our utilization of this data, we presented the models with a series of screenshots with the actions drawn on them, as shown in Figure 3.

C Experiments

Models Configuration For task description prediction, we used the most recent version of Gemini, namely Gemini 1.5 Pro, updated in Vertex AI[2] as of May 2024, with a sampling temperature of 1.0, and GPT-4-Turbo[3] (version gpt-4-turbo-2024-04-09) with a sampling temperature of 0.6. For automatic evaluation, we utilized the latest release from the GPT family, known as GPT-4o.

Data For the manual evaluation, we randomly sampled over 100 data points from the both Android and Web datasets. We then prompted models to predict the user goal. During the evaluation, we conducted a thorough verification process to ensure that both the gold references and the trajectories were of high quality.

We rejected instances where the trajectory did not fulfill the gold reference, which occurred more frequently for Android. Additionally, for web-based tasks, we excluded cases where the screenshot was not rendered properly, making it difficult to interpret the user's action, as well as instances where the bounding box was empty.

As a result of this verification process, 17% of the Android examples and about 5% of the web examples were rejected. Ultimately, as a result of this process, the samples that were manually evaluated in Sections 3.2 and 5 were of high quality and free from such issues. For the automatic evaluation, as described in 5, we utilized the entire test sets from both datasets.

Web We encountered a technical challenge with the Mind2Web data due to the nature of its image captures. Unlike standard viewport captures, which represent the visible area of a web page on a typical screen, the images in Mind2Web had a median height of 4200 pixels, significantly exceeding typical web page dimensions, with 20 percent of the images exceeding 7000 pixels in height. Initial tests showed that these oversized images introduced noise and negatively affected model performance.

To address this, we implemented a heuristic truncation method, reducing image height while ensuring the interaction element remained visible within the truncated image. This was achieved by utilizing the bounding box metadata provided by Mind2Web. Similar to the approach taken by (Zheng et al., 2024) and introduced by (Yang et al., 2023a), we drew the red bounding box around the interaction element to guide the model's attention. Additionally, adding the bounding box helped resolve ambiguities at times that the textual action description is not sufficient. For example, sometimes action descriptions are simply empty, and does not contain any description about the element itself. While in other cases, action descriptions exist, but the textual description matches multiple element descriptions and thus results in ambiguity that is only resolved by drawing the bounding box. For example, a button labeled "Add to Cart" is typically associated with each item in a web shopping list. Without the bounding box, it is impossible to determine which specific button was clicked.

Android In our efforts to replicate experiments from web, we encountered challenges due to differences in data format. Unlike the web, AitW does not provide textual information associated with clicked elements. Instead, it offers x,y coordinates representing the center of the tapped area. This distinction in data structure made it impractical to conduct experiments involving both actions written alongside screenshots.

To address these challenges, we utilized the dataset's utilities to overlay actions on top of the screenshots, as well as bounding box annotations for post-process detected UI elements. However, we found that the added element annotation marks often confused the model added noise and critical information on the screen. As a result, we proceeded with experiments using a sequence of screenshots that had the actions (tap, drag, and type) drawn over them. AitW's provided visualization tools to draw the actions, also included labels of special actions such as the back button, home button, and enter. As well as a special "status" action: either Task Complete or Task Impossible. However, we found that the label Task Complete often confused the models. Despite our efforts to

[2]https://cloud.google.com/vertex-ai

[3]https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4

make the models ignore it, we eventually abandoned this specific annotation as it is a technicality of data representation.

D Error Analysis

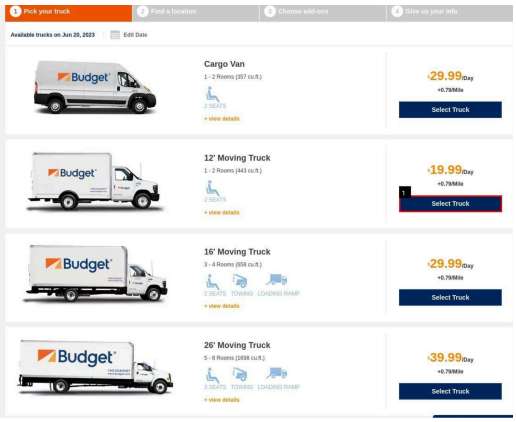

Figure 4: An illustration in which the user chose a specific car primarily for its 12-inch feature, but since it was also the cheapest, annotators incorrectly assumed cost was the deciding factor.

In this section, we aim to provide a more detailed error analysis with respect to model and humans.

Web In our error analysis of 30 mismatched task descriptions, distinct patterns emerged between GPT and Gemini models. GPT frequently wrote tasks as navigational procedures, with 12% of generated tasks starting with "Navigate" (out of the 1,000 predicted tasks). Additionally, over 20% of the manually inspected errors involved misinterpreting the task's intent, often producing broad descriptions lacking crucial details. Conversely, Gemini's errors were typically more fine-grained, often capturing the task's essence but struggling with specific details like dates, numbers, or locations.

Both models also occasionally produced task descriptions that felt artificial, incorporating information a user would be unlikely to know beforehand due to the models' access to the full user trajectory. For example, a task like "Read recent news about Apple stock" might be predicted as "Read the article 'X' about Apple stock" if the model observed the user clicking on a specific article 'X'.

With respect to human annotators, we found that most disagreements between human-generated task descriptions and gold task descriptions resulted from humans making more generalized tasks. This happened because they choose the most natural or probable constraints if no action provides evidence for a less likely constraint. Sometimes, they don't write the constraint or any other one if nothing seems probable. Figure 4 demonstrates such a case, the truck picked by the user is the only 12-inch wheel truck but also the cheapest truck among the listed options, as no prior action gave evident to the 12-inch constraint, both human generated task labeled it as "Book the cheapest truck..." while the gold task description was "Book a 12-inch wheel truck...". Additionally, with respect to task ambiguity, each trajectory in this experiment resulted in three task descriptions, two from the human annotators and one from the gold reference. We calculated the number of trajectories where all task descriptions matched each other and those where they did not. We found that 72% of trajectories had a match among all tasks, 24% had a match between two tasks, and the rest had no matching tasks. Although not exhaustive, this highlights that most trajectories in Mind2Web are probably non-ambiguous.

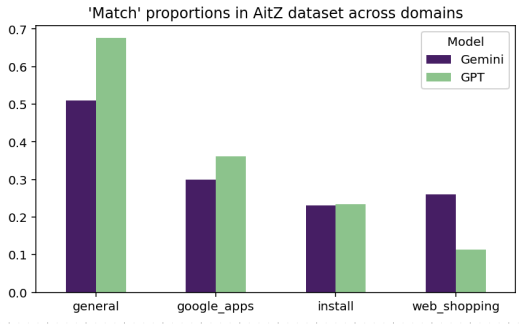

Figure 5: Comparison of "Match" proportions between Gemini 1.5 Pro and GPT-4-Turbo models across the different domains

Android Comparing GPT and Gemini's performance on the Android dataset revealed notable variations across different domains. Both models were proficient in the "General" domain, but faced challenges in "Web Shopping" and "Install". This disparity is due to the nature of the "General" dataset, which contains search queries that explicitly reveal the user's intent. "Google Apps" tasks overlap with PixelHelp, primarily involving settings configuration. Some tasks are ambiguous, such as a button toggle, but the trajectory displays only one option. Additionally, specific tasks request actions that have already been completed, e.g Turn On location history, but the trajectory only shows viewing the Location History setting page, leading to

confusion for models. Additionally, the models faced difficulty comprehending the correct order of the sequence of actions that occurred, frequently mistaking the final state (on or off).

On the "Web Shopping" domain, Gemini provided excessive details about specific products ("Add Razer Kraken X for Console Gaming Headset for PC/PS4/PS5/Xbox/Switch - Black/Blue to cart on Best Buy Canada"). GPT often missed the main purpose of the task and suggested abstract tasks such as: "log into an account" or "Decline the offer to protect a purchase with an insurance plan on a shopping website".

The "Install" domain often presents ambiguous tasks in the format "open (install if not installed)" which confuses both models. Furthermore, in some cases the apps were already pre-installed which made it impossible to predict (model predicts "open"), providing only partial satisfaction in one direction. These results indicate that further refinement and training may be needed to improve the models' performance in specific domains.

For human generated tasks, "General" dataset presented minimal challenges for human annotators. This was attributed to their ability to effortlessly comprehend the user's intended intent based solely on the visible search query. However, analogous to the model challenges encountered, ambiguous tasks within the "Install" dataset proved challenging for humans as well. Conversely, unlike models, humans exhibited impeccable performance in comprehending the final state of the desired setting configuration within the Google Apps domain, if the original goal was specific and not ambiguous. Shopping tasks, on the other hand, posed a distinct challenge for humans. They struggled to grasp the rationale behind selecting an item when the original task was to choose the cheapest or the first result. These findings underscore the multifaceted nature of goal task prediction and emphasize the significance of addressing specific domains.

E Instructions and Prompts

You will be given an observed UI trajectory (a series of actions within a website or app), along with two task descriptions, labeled A and B. Your goal is to provide four annotations for each pair of task descriptions based on the observed trajectory:

1. Is A fulfilled by the trajectory ? (Yes / No)
2. Is B fulfilled by the trajectory ? (Yes / No)
3. Does A satisfy B ? (Yes / No)
4. Does B satisfy A ? (Yes / No)

Definitions:

- Task fulfillment by Trajectory - Task A fulfilled by a trajectory if the trajectory successfully completes the requested action, or if the trajectory provides the information sought by the user.
- Satisfies relation between two tasks - Task A *satisfies* Task B if and only if every reasonable trajectory that fulfills A would also fulfill B. This means completing A necessarily leads to completing B.

Key Assumptions:

- Proficient User - Assume the user is proficient with UIs and familiar with the general structure and functionality of the website/app.
- Fixed UI, Dynamic Content - The website's / app layout, information display, input fields, default values, and terminology are fixed. However, the specific content (e.g., available products, search results) can change.
- Terminology - Terms used within the website/app are considered synonymous. For example, if a clothing site lists both size and garment length in centimeters, these are considered interchangeable.

Instructions:

- Ignore Semantic Equivalents - Disregard differences in wording when A and B are semantically equivalent, including variations in site terminology as mentioned above.
- Ignore Navigation Details - Disregard navigation instructions in the task descriptions like "*open the settings menu to* increase the screen brightness by 10%". Also, ignore mentions of the specific website or app name.
- Relative Dates/Times are OK - Dates and times can be relative (e.g., "in 2 hours") if the current date/time can be inferred from the trajectory. For instance, most sites used for scheduling meetings in a calendar display the current date.
- Partial Actions - Note when one task description asks for a partial action compared to the other (e.g., one asks to buy a ticket, the other only to find information).
- Specific vs. General Instructions - Pay attention to cases where one description is very specific (e.g., "Reserve a Chevrolet Colorado truck") and the other is more general (e.g., "Reserve a truck"). Similarly, an instruction to "search for French chansons and play the first one" might be interpreted as an instruction to play a specific song.

Generally, neither description satisfies the other in these cases. We assume the site is fixed but the content may change, and the user doesn't necessarily know which truck is available or what the first song would be. Therefore, the specific instruction doesn't usually satisfy the general instruction (and vice-versa, of course), as giving those instructions at a different time might yield a different result. However, use your best judgment if it's reasonable to assume the user knows the specific context of the site, and the difference might be just a rephrasing. For example, a user asking to "find out when is the next NBA game" might mean the same as "find out when is the Celtics-Heats match", as they know what is the next game. There is a level of subjectivity here in determining what is reasonable to assume the user knows, and we expect you to use your best judgment in these cases.

Figure 6: Instructions for human-annotators to conclude if a task is fulfilled by a trajectory, and if two task descriptions satisfying each other. These instructions were the core prompt of the automatic evaluator.

You will be given a trajectory of UI actions performed by a user, and your task is as follows:

1. Most likely goal - Write the most likely goal that led the user to perform this trajectory.
2. Notes - We ask you to also write notes, with respect to challenges you faced during the writing, decisions you made, assumptions you take etc...

General Guidelines:

- Standalone goals - Write standalone goals. Do not write goals that are contextualized by the website or depend on memory or other personalizations (location etc...). Assume no prior knowledge of the website's content. Keep in mind that the user's were not familiar with the content but were familiar with the purpose of the app / website and its functionality.
- Non-time relative goals - For web write goals that are not relative to time and ensure you write full dates when possible. For example, write 'Reserve restaurant for 1.6.2024' (Or any other date format you like) instead of 'Reserve restaurant for next week'. For Android you can use relative time ranges if you find it reasonable.
- Imperative goals - Write imperative goals that give commands or requests, not goals that ask questions or make inquiries.
- Default values - Use your common sense to judge if a default value should be present in the goal description or not. Generally, here are few additional points that should provoke your thoughts when considering a default value:
  - Think about the inverse task, i.e you have a goal in mind and only then you perform the task on the web. Which default values are reasonable that you would have in mind ?
    For example, dates when booking something / searching for flight tickets etc.
  - Consider default values that are transferable across similar websites. For example, if you book a flight on UNITED and they have a default setting related to membership, this might not be applicable to DELTA airlines.
  - Assess the importance of the default value in completing the task without additional assistance. For instance, many registration forms have a pre-checked checkbox to send notifications, which is not essential to the task, so less likely that the original goal included this default.
- Starting Point - For Android assume all trajectories start at the homescreen, for web, it is in the particular website.
- Avoid mentioning explicit apps unless you find it necessary - 'Check calendar' and not 'Check Google calendar'.
- History - In Android, many trajectories have pre-existed history (from previous sessions) . Be aware of it, and judge when you need to incorporate this in the goal and when not.

Figure 7: Goal identification annotation task instruction to measure human performance.

Input Structure:
You will receive a series of screenshots and actions representing a user's interactions on a website / application. Each item contains an action performed by the user and a screenshot depicting the state of the website before action execution. Below is a more detailed description of the action format structure and screenshot.

Mind2Web Section
Actions are formatted to clearly identify the type of user interaction with web page elements. The format specifies the element type, a detailed description, and the interaction performed. This setup allows for precise specification of interactions, particularly noting that TYPE and SELECT actions require additional text values:

General Format: [element type] element description (usually the text of the element) -> Operation (TYPE / CLICK / SELECT): action-specific details (For TYPE / SELECT only).
1. TYPE: Inputting text, requires specifying what text is typed (e.g., [combobox] Departure station, London selected. -> TYPE: Edinburgh).
2. CLICK: Activating or selecting an element, no additional text required (e.g., [span] Edinburgh (Waverley) -> CLICK).
3. SELECT: Choosing from a dropdown or list, requires specifying the selected option (e.g., [listbox] Hour -> SELECT: 17).

Each screenshot shows the website state just before the corresponding action occurs. These screenshots often include a RED bounding box highlighting the element of interaction. The screenshot serves as a visual context, showing details not captured by the action description..

AitW Section
User actions are indicated by blue symbols within the screenshot, marked by a plus sign (+) for tapping and an arrow for scrolling.

Your Task:
Analyze the input sequence to deduce the user's underlying objective that prompted these actions. Utilize the screenshots to gain insights into the user's intentions, focusing on elements highlighted or implicated by the actions.

General Guidelines:
1. Imperative Format: Sentences must be structured in the imperative form, directly issuing commands without stating the subject performing the action.
2. Action-Oriented: Begin the sentence with the verb that denotes the action to be taken, ensuring that it directly addresses the desired outcome.
3. Specificity and Detailed: Include all pertinent details necessary to complete the action without overloading the sentence with unnecessary information.
4. Clarity and Conciseness: Aim for straightforwardness and brevity, avoiding any ambiguity. The instruction should be easily understandable at a glance, making it actionable without requiring further clarification.
5. Ignore advertisements, pop-ups, and default dialogs (e.g., "Accept cookies," "Take me to Gmail").
6. Do not specify the app/website or platform.

Output Format:
"step-by-step description": "Provide a numbered list where each entry corresponds directly to a specific screenshot, detailing the user's actions and the visual context provided by the screenshots.",
"concise task": "Summarize the user's overall goal that motivated the sequence of actions based on the step by step description."

Figure 8: Instruction used to guide GPT and Gemini when predicting a task description given a UI trajectory. We swap AitW section and Mind2Web according to the input.

You're given two sentences, A and B, representing user goals on a website or app, and a trajectory for Task A.

Your Task: Determine if A satisfies B.

Definitions:

- **Fulfillment:** A trajectory fulfills a task if it presents information satisfying the user's query (even with extra info). For transactional tasks, it must complete the requested operation.
- **Satisfaction** A satisfies B if EVERY trajectory fulfilling A also fulfills B. This means completing A logically leads to completing B.

Analysis Steps:

1. **Extract Requirements:** List requirements and constraints of A and B in a table, noting if A satisfies each B requirement. Include action verbs (e.g., "find," "book"). If any answer is no, A doesn't satisfy B.
2. **Rigorous Comparison:** Is each requirement in B met by A? If A has stricter requirements, it satisfies B. If A is looser, go to step 3.
3. **Counter-Example (Optional):** Find a case where requirement A is met while requirement B is not, considering dynamic content changes like prices and availability. Ignore static content like structure and checkboxes.
4. **Chain of Thought:** Clearly explain your reasoning, referencing the requirements. Explain how each requirement in A is logically fulfilled by B or how a counterexample shows it doesn't.
5. **Decision:** Write your answer as: [SATISFACTION] YES/NO [/SATISFACTION]

Key Considerations:

- **User Goals:** Focus on the user's underlying goals, not just literal words.
- **Default Values:** If a requirement is met by a default value that is present in the website/app interface, it can be omitted from the constraint list.
- Prioritize logical reasoning over assumptions.
- **Dynamic vs. Static Content:** Consider if content changes often (e.g., top movies) or stays the same (e.g., filtering by user's rating). If A needs dynamic content and B static, A doesn't satisfy B.
- **Direction:** Only answer if A satisfies B (A -> B), not the reverse.
- **Relative vs. Specific Times:** When comparing relative times, such as "tomorrow," and specific times, such as "April 15th," it is acceptable to disregard the relative times or assume that they are acceptable.
- **Time Ranges and Dates Matter:** Time ranges (e.g., "tomorrow" vs. "the week of April 15th") are considered distinct. Explicit different dates are considered distinct.
- **Default Values:** If A misses default values, but the trajectory has them, consider A satisfying B with respect to this requirement.
- Use the Trajectory: If A and B are unclear, the trajectory may show they refer to the same action or process.
- **General vs. Specific:** A specific task A can satisfy a more general task B if completing A fulfills B's requirements. If A is specific and B is general, A might satisfy B, but it depends on whether all conditions in B are fulfilled by A.
- If A is general and B is specific, A does not satisfy B.
- In the context of this task, we will treat the terms "buy," "add to cart," and "check out" as interchangeable.

Determine whether A satisfies B with the given trajectory. Input:

Figure 9: Model instructions for evaluating Satisfaction relation between two task descriptions (A and B), given a corresponding trajectory. Includes web/mobile data format instructions and few-shot examples for experimentation (detailed on the next page).

Example 1: Stricter A Satisfies Looser B

- **A:** Purchase a one-way ticket from NYC to LAX **on Delta Airlines.**
- **B:** Book a flight from New York to Los Angeles.

Analysis: **A:** Purchase flight, one-way, NYC to LAX, Delta Airlines. **B:** Book flight, New York to Los Angeles. A's requirements are a subset of B's. Purchasing a one-way flight from NYC to LAX operated by Delta Airlines, will fulfill B requirements: Booking flight, NY to LA. [SATISFACTION] YES [/SATISFACTION]

Example 2: Default Values & Satisfaction

- **A**: Order a pepperoni pizza
- **B**: Order a **large** pepperoni pizza
- **Trajectory:** Website defaults are set to large size when ordering a pizza.

Analysis: **A:** Order, Pizza, pepperoni. **B:** Order, Pizza, Pepperoni, Large Size. B has stricter restrictions for a Large pizza size, but the trajectory shows the default pizza size is Large. A satisfies B because the default size fulfills the requirement. Any reasonable trajectory executing A, ordering a pizza (not requiring setting a specific pizza size) will satisfy B. [SATISFACTION] YES [/SATISFACTION]

Example 3: Dynamic Content & Non-Satisfaction

- **A:** Buy the **cheapest** flight to London.
- **B:** Book a flight to London for **under $500.**
- **Trajectory**: Shows the cheapest flight is currently $450.

Analysis: **A**: Buy, flight, **cheapest**, destination: London. **B:** Book, flight, **under $500**, destination: London. Executing A doesn't guarantee satisfying B, as prices can change. Counter-example: The price could rise above $500, fulfilling A but not B. [SATISFACTION] NO [/SATISFACTION]

Example 4: Website Structure and Satisfaction

- **A:** Find a **5-star** hotel in Rome.
- **B:** Find a **highly-rated** hotel in Rome.
- **Trajectory**: Shows a filter for star ratings, with 5-stars being the highest option.

Analysis: **A:** Find, **5-star,** a hotel, Rome. **B:** Find, **highly-rated,** a hotel, Rome. Since the website structure doesn't allow for higher ratings than 5, A satisfies B. [SATISFACTION] YES [/SATISFACTION]

Example 5: Looser A Fails to Satisfy Specific B (Counter-Example)

- **A:** Book a room in a hotel **with a pool** in Paris.
- **B:** Reserve a room at the **Hotel Ritz** in Paris.
- **Trajectory**: User is booking, Ritz hotel in Paris. Trajectory shows Ritz hotel has a pool.

Analysis: **A:** book, room, **any hotel, has a pool**, Paris. **B:** Reserve, room, **Hotel Ritz,** Paris.
A's requirement for a hotel with a pool is looser than B's requirement for Hotel Ritz. Counter-example: The user could choose a different hotel in Paris that has a pool, still fulfilling A but it won't fulfill B. [SATISFACTION] NO [/SATISFACTION]

Example 6: Stricter verb in A Satisfies Looser verb in B

- **A: Sign-up** for a Fastbreak program.
- **B: Find** information about the Fastbreak program.

Analysis: **A: Sign-up**, Fastbreak program. **B: Find** information, Fastbreak program. A's requirements inherently include B's. Signing-up for something necessitates finding information beforehand about this thing. [SATISFACTION] YES [/SATISFACTION]

Example 7: Looser verb in A Fails to Satisfy a Specific verb in B (Counter-Example)

- **A: Find** information about the Fastbreak program
- **B: Sign-up** for a Fastbreak program.

Analysis: **A:** Find information, Fastbreak program. **B:** Sign-up, Fastbreak program. A is less strict than B. A counterexample is simply finding the necessary information without enrolling in the program. [SATISFACTION] NO [/SATISFACTION]

Figure 10: Few-Shot exemplars demonstrating the Satisfaction Relation Prompt (Figure 9). These simplified examples highlight the nuances of task satisfaction in real-world scenarios.