Small Models, Big Results: Achieving Superior Intent Extraction through Decomposition

Danielle Cohen[*1], **Yoni Halpern**[*1], **Noam Kahlon**[1], **Joel Oren**[1],
Omri Berkovitch[1], **Sapir Caduri**[1], **Ido Dagan**[†1,2], **Anatoly Efros**[†1]

[1]Google, [2]Bar-Ilan University

Correspondence: daniellecn@google.com, yhalpern@google.com

Abstract

Understanding user intents from UI interaction trajectories remains a challenging, yet crucial, frontier in intelligent agent development. While massive, datacenter-based, multi-modal large language models (MLLMs) possess greater capacity to handle the complexities of such sequences, smaller models which can run on-device to provide a privacy-preserving, low-cost, and low-latency user experience, struggle with accurate intent inference. We address these limitations by introducing a novel decomposed approach: first, we perform structured interaction summarization, capturing key information from each user action. Second, we perform intent extraction using a fine-tuned model operating on the aggregated summaries. This method improves intent understanding in resource-constrained models, even surpassing the base performance of large MLLMs.

1 Introduction

Advancements in the capabilities of multi-modal large language models (MLLMs) has led to recent interest in modeling sequences of user interactions with phone and web graphical interfaces, both for the purposes of automation (Wang et al., 2024b; Jiménez-Ramírez, 2024), and understanding (Berkovitch et al., 2025; Zhang et al., 2025).

In this work, we focus on the user intent extraction task, which consists of producing a free-form description of the inferred intent of a user from a sequence of interactions with a device.

Large MLLMs are naturally fairly good at this task, however, it is more challenging for smaller models (E.g., Gemini 1.5 Flash 8B (Gemini Team et al., 2024) or Qwen2 VL 7B (Wang et al., 2024a)). The performance of smaller models is important for user interaction tasks due to their ability to operate within a private, on-device environment like a phone or browser, with reduced cost, energy usage, and latency (Xu et al., 2024).

In this paper, we introduce a two-stage approach for extracting user intent with small models. In the first stage, each atomic interaction is summarized. In the second stage, the full sequence of summaries is fed to a second model which outputs an intent. The overall flow is illustrated in Figure 1. Using semantic equivalence metrics on public UI automation data, our two-stage approach demonstrates superior performance compared to both smaller models and a state-of-the-art large MLLM, independent of dataset and model type. Our approach also naturally handles scenarios with noisy data that traditional supervised fine-tuning methods struggle with. The modular nature of the architecture is helpful from an engineering perspective, allowing us to evaluate the approach in detail and identify key areas to improve.

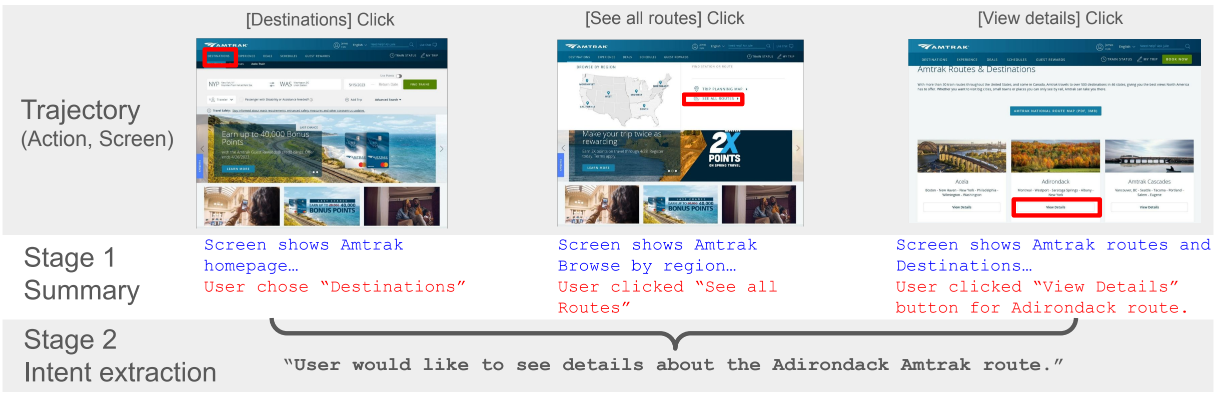
Our contributions can be summarized as follows: 1. We describe an effective decomposition of intent-extraction that unlocks the potential of small models; 2. We present non-trivial design components related to each stage of the decomposition; 3. We extensively evaluate our approach and demonstrate the effectiveness of our method across a range of data sets, base models and metrics.

2 Background

2.1 Intent Extraction from UI Interactions

We formalize the intent extraction task, sometimes called goal understanding, similarly to prior works Berkovitch et al. (2025) and Zhang et al. (2025). Consider a user journey T within a mobile or web application, represented as a sequence of interactions: $T = (I_1, I_2, ..., I_n)$, where each interaction $I_i = (O_i, A_i)$ consists of an observation, O_i, and the action, A_i, the user performed at that step. This description is general and different modeling ap-

[*]These authors contributed equally.
[†]co-senior author.



Screen shows Amtrak homepage… User chose "Destinations"

Screen shows Amtrak Browse by region… User clicked "See all Routes"

Screen shows Amtrak routes and Destinations… User clicked "View Details" button for Adirondack route.

"User would like to see details about the Adirondack Amtrak route."

Figure 1: Proposed intent extraction flow (Described in detail in Section 4). Individual interactions (input as action strings and screenshots) are summarized and then the summaries are combined to output a short inferred intent for the trajectory. The summarization step uses both action strings and visual screen information to output a structured summary with two fields corresponding to screen summary (top, blue) and user action (bottom, red).

proaches have used different representations for observations and actions (e.g., textual descriptions, screenshot images, DOM hierarchies, etc.) (Rawles et al., 2023; Burns et al., 2022). The objective of the intent-extraction task is to generate a free-form sentence describing the user's intent. Effectively, this setting can be thought of as the inverse problem of the UI automation task, with inputs and outputs swapped. Rather than producing a sequence of actions from an instruction, we ask "what was the user trying to accomplish with this trajectory?". Intent extraction has been identified as an important building block for UI automation tasks, proactive assistance, and personalized memory (Berkovitch et al., 2025; Zhang et al., 2025).

Very recently, a few works have begun addressing intent extraction from UI interactions. Berkovitch et al. (2025) proposed this novel task, and seemingly were the first to point out that it can be viewed as the inverse task of UI Automation. They evaluated MLLM intent-extraction performance over UI automation datasets (swapping input/output roles). As input, they considered screenshot images and textual descriptions of user actions, as we do in our work, and assessed the performance of standard MLLMs using a fairly simple prompting approach. In our work, we follow this evaluation approach, while testing an improved version of their prompt as a baseline. Zhang et al. (2025) also evaluated their SummAct model over UI automation datasets, but took as input *only* short textual descriptions of the specific UI elements with which the user interacted and the respective user actions, without considering the global screen context, as we do. Loosely similarly to our method, they decomposed intent generation into two consecutive stages, though these are substantially different than ours and required intervening with the LLMs attention mechanism for fine-tuning (see Appendix H for detailed comparison of the two methods). Finally, the UI-JEPA model (Fu et al., 2025) takes as input videos of the entire UI session. Their method adapts a specialized video-embedding method to work with videos of UI sessions, and then finetunes an LLM decoder that generates intent descriptions based on these embeddings. Overall, SummAct, our work, and UI-JEPA consider different types of inputs, providing increasingly richer contexts with increasing complexity: "local" text descriptions of the user action and its respective UI element, accounting for the full screen image, and processing a complete video of the entire session (respectively). It is left for future work to thoroughly explore the pros and cons of these alternative inputs under different scenarios.[1]

With respect to evaluating model-predicted intents, a good intent is *faithful:* only describes things that actually occur in the trajectory; *comprehensive:* provides all of the information about the user intent required to re-enact the trajectory; and *relevant:* does not contain extraneous information beyond what is needed for comprehensiveness. However, even with a well-defined ground truth intent, accurately evaluating a model's extracted intent is challenging. User intents often contain many details, such as trip planning specifics or transaction

[1]An empirical comparison with SummAct on equal grounds was not possible, since their code was released without full prompts close to the submission deadline. Similarly, the UI-JEPA inference code hasn't been released yet while the license on the dataset restricts our lab's usage.

data, which require metrics that can handle partial matches. Such metrics fall into two categories: semantic, which analyze underlying meaning, and lexical, which assess surface-level word overlap. As Caduri et al. (2025) show, lexical metrics (e.g., BLEU and ROUGE) correlate poorly with human judgments of intent similarity, as they merely compare words. In contrast, semantic metrics, such as NLI (Natural Language Inference) and BI-Fact (a bi-directional variant of FActScore (Min et al., 2023)), strive to capture the intended meaning.

Further, intent extraction is inherently subjective, as a single trajectory could have been driven by multiple underlying motivations (e.g., a user may have selected a flight based on its price versus its departure time). This subjectivity is evident in prior work, such as Berkovitch et al. (2025) where human-composed intentions matched each other in only 80% and 76% of web and phone trajectories, respectively. This level of human agreement may be considered a practical upper bound for performance on this task.

2.2 UI Interaction Datasets

Recently, a number of datasets have been developed for evaluating UI interaction agents, (surveyed in Wang et al. (2024b)). We use two that are representative and suitable for measuring the intent extraction task. We confirmed that our usage of the data adhered to all ethical and legal standards.
Mind2Web (CC BY 4.0 license) (Deng et al., 2024): Has 2,350 human demonstrations on websites. Each user trajectory is on average 7.3 steps long and contains screenshots and actions for each step, as well as a high level description of the task the human was asked to perform.
AndroidControl (Apache 2.0 license) (Li et al., 2024): Has 15,283 examples of humans performing tasks on Android apps. Each user trajectory is on average 5.5 steps long, and contains screenshots and actions for each step, as well as a high level description of the goal.

Mind2Web's data collection included a validation step where annotators verified the alignment between the completed trajectory steps and the intent, making this dataset highly suitable for the intent extraction task as well. This crucial step was absent from the AndroidControl collection protocol, resulting in noisier labels. For example consider the following task "Delete all emails from sender X" in a scenario where there were no emails from that sender. Based on the execution of task it is impossible to identify that the original goal was to delete emails. We preprocess labels to remove clearly irrelevant statements (Section 5.2) and analyze the effect of remaining discrepancies between the labels and trajectories in Section 6.2.

2.3 Related Research Lines

User interaction understanding for HCI Single screen summarization as a special case of image description has been extensively studied for the purposes of e.g., accessibility, automation, and question answering (e.g., Li et al., 2021b; Bai et al., 2021; Li and Li, 2022; Wang et al., 2021; Yang et al., 2025). Our setting of identifying and summarizing intents from trajectories has been recently proposed in Berkovitch et al. (2025); Zhang et al. (2025); Jiménez-Ramírez (2024).

Multi-stage summarizations Decomposing a complex task into smaller simpler stages is a well-known approach for problem solving. Hierarchical models are common in summarization tasks of many modalities, e.g., text (Christensen et al., 2014), audio (Li et al., 2021a), video (Zhao et al., 2022; Cheng et al., 2024). Chain of Thought (CoT) reasoning (Wei et al., 2022) is a popular general-purpose prompting method to decompose a problem into smaller parts. Khot et al. (2022) propose an automated decomposition step that delegates different parts of the problem to distinct model calls.

3 Baseline Modeling Approaches

In this section, we first present natural baseline approaches for addressing our task, whose lessons led us to develop our decomposed two-stage approach which will be described in Section 4. Our task is a text generation task, where intent descriptions are generated from the multi-modal input of UI trajectories. As such, it is most natural to address it through multi-modal LMs, applying either prompt-based or fine-tuned methods. The focus of our work is to explore the use of small LMs, for eventual utilization on-device. The particular models used are specified in Section 5.1, including a top-tier large model as a reference point.

Prompt-based methods Such methods are advantageous in that they do not require training data, instructing a *generic* LM via its prompt. We found that a CoT prompt worked best. Specifically, our CoT prompt (see I1) instructs the model to first generate a sequence of individual descriptions of the user intents within *each UI interaction*, and then to consolidate these interaction-level description into the final description of the accumulated user intent along the trajectory.

Fine-tuned models Since performance of prompting a generic model may not be fully aligned with the intended task output and prompt-based performance of small LMs might generally be limited, we also explore baseline fine-tuning methods. To that end, we fine-tuned small models using available training datasets, specifically those developed for the inverse problem of UI automation, while swapping their input/output roles (see Section 2.2).

Both prompt-based and fine-tuned baselines require large context window to contain the entire user trajectory including images. As described in Section 5.1, practically this required some filtering over the input to fit the available context size.

4 A Decomposed Two-stage Model

While CoT prompting works well with large language models (LLMs), we observe limitations in both CoT and fine-tuned small LMs when presented with the full trajectory. When applying CoT reasoning, small models struggle to generate high-quality thoughts that cover the full trajectory. Fine-tuned small models also have trouble generating comprehensive intents from the full trajectory.

These observations led us to develop a decomposed, two-stage approach that emulates the CoT process, illustrated in Figure 1. First, we use prompting to generate a summary for each interaction (consisting of a visual screenshot and textual action representation) in a trajectory. This stage is prompt-based as there is currently no training data available with summary labels for individual interactions. Second, we feed all of the interaction-level summaries into a second stage model to generate an overall intent description. We apply fine-tuning in the second stage and we describe that process in more detail below (Section 4.2). The following subsections provide a detailed description of each stage in our proposed method.

4.1 Interaction Summarization

In the first stage, we summarize each individual user interaction $I_i = \{O_i, A_i\}$ of the length-n trajectory $T = (I_1, \ldots, I_n)$. The summarization uses visual and textual information to extract relevant information regarding the user's goals and actions within that interaction. The output of this stage is a summary of the screen context and user action (see Figure 1). This key information, which describes this particular user interaction, will be used in the subsequent fusion stage. This summarization process is entirely prompt-based (see I3).

We add the two following improvements to the design of this stage, which improves overall performance, as shown in ablation studies in Section 6.3.

Context window While the primary task is to understand I_i in isolation, we recognize that often context can be crucial for eliminating ambiguity and/or uncertainty. Therefore, in addition to I_i the model also receives as input the preceding and successive interactions, I_{i-1} and I_{i+1}, respectively. This allows the model to use e.g., the visual cues from both the current the next screenshot to understand the user action at step i.

Structured summaries We request that the summary be structured in two distinct components: (a) the relevant screen context – a short list of salient details on the current screen O_i, and (b) the user action: a list of mid-level actions that the user took in the current interaction (example in Figure 1). Despite being structured in two fields, the visual cues from the screenshot are also used to understand the action (e.g., in Figure 1, the visual cues are helpful to extract that the click relates to the "Adirondack" section of the page.) As a practical measure for dealing with cases where the model outputs its (unwarranted) interpretations of the user's underlying intent, we also instructed it to output those in a third field (labelled "*speculative intent*") that we discard before proceeding to the next stage.

This structured format was selected to address challenges encountered with alternative prompting strategies. Simply asking the model to be concise resulted in summaries that lacked crucial details. Conversely, prompting for comprehensive information, including user intent, led to excessive speculation that could hinder the subsequent summary fusion stage. Our structured format aims to capture a broader range of information while enabling the removal of speculative elements prior to the second stage. This balanced approach mitigates the risk of contradictions and improves the overall summarization process.

4.2 Generating Session-Level Intent

In the second stage, we aggregate the information extracted during the first stage. A second-stage model takes as input the summaries of all interactions in the trajectory to infer the user's overall intent. This aggregation stage is implemented by fine-tuning a model to specialize in the aforementioned aggregation. For fine-tuning, the training data consists of: a. input summaries representing all interactions in the trajectory, and b. a corresponding ground truth target that describes the user's overall intent in the given trajectory. For comparison, a variation with no fine-tuning, which is fully prompt-based, is also available as part of our ablation study in Subsection 6.3.

We noted in early explorations that naively applying fine-tuning yields a model that embellishes or hallucinates by introducing details that were not present in the screen summary inputs. On further examination, we found that the training procedure encourages the model to act this way since the inputs are potentially incomplete summaries and the targets are the complete intent statements. Thus, when looking at (input, target) pairs, the model learns that it needs to sometimes add additional information in order to produce the target intent.

Following this insight we refine our target intents at training time to remove details not reflected in the corresponding input (using a large language model, see Figure I5 for details on the prompt used in this stage). This ensures that the model will learn to infer intents based solely on the provided interaction summaries. We discuss the effects of this cleanup stage in Subsection 6.3.

5 Experimental Setting

5.1 Models

We focus on smaller, multi-modal models, that can be fine-tuned. In particular, we use *Gemini*[2] *1.5 Flash 8B* (Gemini Team et al., 2024) and *Qwen2 VL 7B* (Apache 2.0 license) (Wang et al., 2024a). For comparisons with a MLLM, we use *Gemini 1.5 Pro* (Gemini Team et al., 2024).

When using Qwen2 VL 7B for baseline models, we dropped frames randomly from the trajectory if they exceeded the context window length. We found that limiting trajectories to 15 steps was sufficient to run our experiments. We also downsized AndroidControl images by a factor of 4 in each dimension when inputting them to Qwen models. Details of fine-tuning can be found in Appendix B.

[2]Terms of service: https://ai.google.dev/gemini-api/terms

5.2 Datasets and Preprocessing

We use the Mind2Web (Deng et al., 2024) and AndroidControl (Li et al., 2024) datasets as representative user interaction datasets. We follow the standard train/test split of each dataset, fine-tuning with train, and reporting results on test data.

In Mind2Web, we represent the action from the dataset textually: (e.g., "[element name] click" or "[element name] hover".). In AndroidControl, we use the accessibility tree to convert the screen coordinates of the interacted item to an element name and format the action in the same way. In both datasets, we use screenshots as observations. We highlight the interacted element in the screenshot with with a red box (Zheng et al., 2024; Yang et al., 2023). To improve the evaluation of user intent interpretation, goal labels from the datasets were cleaned and restructured to separate platform-specific information from the core intent, see Appendix A for more details.

5.3 Evaluation Metrics

We measure quality of extracted goals using two different semantic equivalence metrics.

T5 NLI (Honovich et al., 2022): A T5-XXL model[3] trained for NLI (Natural Language Inference). We compute the entailment probability of the produced summary from the gold standard and vice versa, and then average the two values to get a single bidirectional score.

BiFact (Caduri et al., 2025): A bidirectional variation of FActScore (Min et al., 2023) developed for assessing the equivalence of intents in UI interactions, demonstrating the highest correlation with human judgments compared to existing methods. This metric deconstructs both the ground-truth and predicted intents into their fundamental factual components using an LLM (we use Gemini 1.5 Pro for this). These components are then compared to measure the extent of coverage. We use the BiFact measures of precision (the proportion of facts in the predicted intent that are present in the true intent), recall (the proportion of facts in the true intent that are captured by the predicted intent) and F1. To assess the robustness of our method, we applied the BiFact metric with an alternative model not used in our experiments. The results remained consistent (Appendix E).

[3]Available at: `https://huggingface.co/google/t5_xxl_true_nli_mixture`

We believe that BiFact, which uses a fine-grained, fact-level comparison, is ideally suited for our task since intents can be composed of many parts (e.g., book a flight, flight is to LAX, flight is on Friday). NLI, which holistically evaluates logical entailment of the full sentences is less ideal, but provides an extra signal.

6 Experiments

6.1 Evaluating Extracted Intents

To show that our decomposed approach is generally helpful compared to baselines across models and data modalities, we evaluate the metrics in Section 5.3 on two different datasets using two different models. The results are displayed in Table 1.

In this table, CoT (Chain of Thought) and E2E-FT (End-to-End fine-tuned) represent the natural baselines described in Section 3. Of these two baselines, neither is uniformly more effective across all settings. On the Mind2Web dataset, which has cleaner labels (described in 2.2), Gemini, as a stronger base model, has higher BiFact F1 and Bi-NLI scores with CoT, whereas Qwen2 VL 7B benefits from fine-tuning. Gemini 1.5 Pro CoT is presented as a comparison to a top-tier large MLLM. We find that on Mind2Web, the fine-tuned decomposed approach allows the Gemini Flash 8B to even exceed the performance of the Gemini 1.5 Pro model using CoT. On AndroidControl, the scores are comparable.

The BiFact score is non-deterministic as it uses an LLM to compute the score. We observe a 0.016 standard deviation on repetition. A more detailed breakdown of performances on the test sets by held-out data type can be found in Appendix F.

Manual verification - Human preference: To further verify, a human rater compared 20 Mind2Web trajectories with intent predictions from Gemini Flash 8B, choosing between CoT and Decomposed-FT responses (details in Appendix D). Overall, Decomposed-FT was preferred in 12 instances, CoT in 4, and 4 were rated equally.

6.2 Label Quality and Comparison with Expert-Written Intents

To understand the quality of the labels after preprocessing (described in Section 5.2 and Appendix A), we elicited expert-written intent statements for 100 examples in the AndroidControl dataset following the annotation protocol in (Berkovitch et al., 2025). In Table 2 we compare the BiFact F1 metric for proposed intents against dataset labels and against expert written intents (more detailed metrics in Appendix Table 5).

Overall, the performance of each model improves when compared to expert annotations, except for the E2E-FT model, which was trained on the noisy labels. The fine-tuned decomposed approach also uses fine-tuning, and could have been expected to similarly suffer from training on noisy labels, but instead it significantly improves when evaluated using expert intents. We believe this is due to our approach to constructing fine-tuning labels (Section 4.2) which removes information present in the gold labels but absent from summaries. Interesting to notice that after cleaning the AndroidControl data, the performance of Gemini 1.5 Pro CoT is similar to the performance on Mind2Web suggesting the gap in performance between datasets is mainly the result of data noise.

6.3 Ablation Study

We consider four variants of our method to estimate the impact of each design choice. The performance of each of these ablations can be found in Table 3.

No Context In this variant, Stage 1 is provided with only a single interaction, without previous or next interactions. Our analysis reveals that incorporating information from the previous and next interactions significantly helps the model to infer the user action in the current screen, thereby leading to a noticeable increase in Stage 1 recall.

Unstructured Interaction-level Summaries Our method instructs the model to output interaction summaries that are structurally broken down into context, user actions, and a speculative intent list (which is removed prior to proceeding to the next stage). Instead, we permit free-form summaries, and the concatenation of those are provided to the goal extraction. Instructing the model to output these particular structured responses allows the Stage 2 model to focus on user actions on the one hand, while mitigating Stage 1 hallucinations as much possible. We notice a slight decrease in both precision and recall, as a result of eliminating this part in our method.

No Fine Tuning In this ablation, the second stage of our model was not subjected to fine-tuning and operated solely on a prompt-based approach. Our findings indicate that this configuration led to a marked decrease in precision. Without fine-

	Mind2Web				AndroidControl			
		BiFact		Bi-NLI		BiFact		Bi-NLI
Method	F1	Precision	Recall		F1	Precision	Recall	
Gemini Flash 8B								
CoT	0.659	0.758	0.647	0.326	0.594	0.628	0.660	0.302
E2E-FT	0.653	0.676	0.671	0.311	0.611	0.655	0.656	0.343
Decomposed-FT	**0.752**	**0.814**	**0.746**	**0.391**	**0.630**	**0.664**	**0.688**	**0.350**
Qwen2 VL 7B								
CoT	0.563	0.694	0.551	0.272	0.538	0.589	0.603	0.280
E2E-FT	0.610	0.670	**0.621**	0.233	0.506	0.594	0.546	**0.343**
Decomposed-FT	**0.623**	**0.736**	0.609	**0.300**	**0.608**	**0.661**	**0.646**	0.333
Gemini-1.5-Pro								
CoT	0.730	0.773	0.745	0.331	0.634	0.612	0.767	0.347

Table 1: BiFact and Bi-NLI results on the Mind2Web and AndroidControl datasets using Gemini 1.5 Flash 8B, Qwen2 VL 7B, and Gemini 1.5 Pro. Best scoring method for each model is bolded. F1, precision, recall are micro-averaged over the dataset.

Method (Gemini-1.5 Flash 8B)	Expert Labels	Dataset Labels
CoT	0.652	0.580
E2E-FT	0.590	0.565
Decomposed-FT	**0.701**	**0.596**
Gemini-1.5-Pro CoT	0.724	0.635

Table 2: A comparison of BiFact F1 scores for intent prediction on the AndroidControl dataset, using expert annotations and dataset labels as ground truth. A more detailed table appears in Appendix C.

Method	F1	Precision	Recall
Decomposed-FT	**0.752**	**0.814**	0.746
- No context	0.711	0.794	0.698
- Unstructured	0.731	0.791	0.737
- No fine-tuning	0.724	0.719	**0.802**
- No label refine	0.738	0.756	0.773

Table 3: Ablation study on Mind2Web using BiFact scores. The Decomposed-FT model is the full model and then each subsequent line shows the effect of removing a single design component.

tuning, the model tended to be more verbose, resulting in a higher proportion of irrelevant or incorrect information being generated. Conversely, this same verbosity contributed to an increase in recall, as a broader range of potential information was captured. However, when considering the F1 score, which balances precision and recall, the fine-tuned version of Stage 2 demonstrated superior performance, underscoring the benefits of the fine-tuning process. For completeness, an analysis of this prompt-based approach on larger models is provided in Appendix G.

No Label Refinement Recall that label refinement was added to address Stage 2 hallucinations. In this variant, we exclude the label refinement step, during the data preparation for the fine-tuning of the Stage 2 model, as described in Subsection 4.2. As expected, after removing this step, we notice a significant decrease in precision. However, we also see a slight increase in recall, suggesting potential areas for improvement in the refinement process.

6.4 Manual Error Analysis

To gain a deeper understanding of the errors produced by the decomposed-FT model, we manually analyzed 20 examples.

Counts are indicated in parentheses after each error type. Some examples exhibited multiple error types, so the counts do not necessarily add up to the total number of examples.

Incorrect screen understanding (6) Includes instances where the model misinterpreted the UI elements or incorrectly understood the user action.

Summary omissions (6) Includes instances where the model failed to capture important on-screen details, like omitting the destination on a travel site.

Hallucinations (4) Includes instances involving generating information not present on the screen, such as claiming the user selected a specific item when they did not.

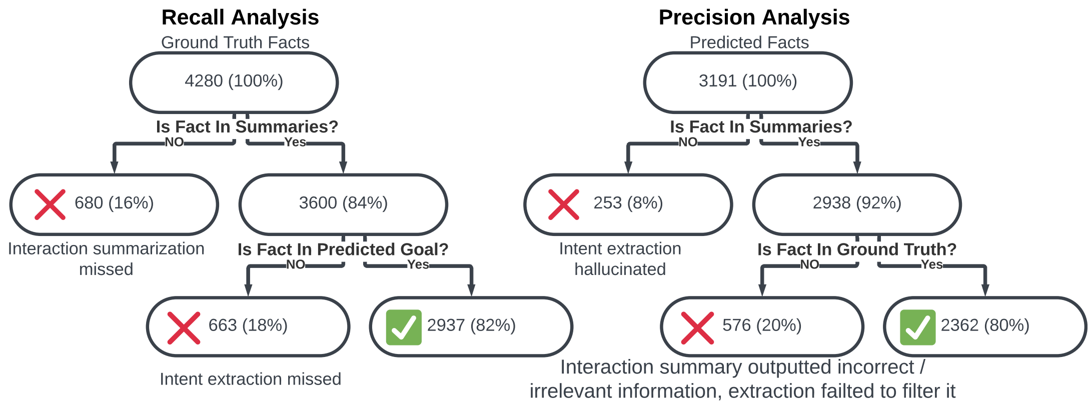


Figure 2: Error propagation analysis of our method on the Mind2Web dataset, tested with Gemini Flash-8B, tracking ground-truth and predicted facts to obtain stage-level recall and precision.

Irrelevant details (0) Includes instances where the model included correct but excessive information. While this error was not present in our full model, it was significant in the "no formatting" models used in the ablation study (Section 6.3).

Intent extraction omissions (8) Includes instances where the second stage failed to include important details present in the individual summaries.

Evaluation Errors (1) These errors were infrequent and typically involved situations where complex screen understanding was required to determine the equivalence of intents.

The majority of issues occur in the Interaction Summarization stage, suggesting potential benefits from distillation training of this stage. Initial experiments, showed no significant improvements from distillation. Further investigation is warranted.

6.5 Error Propagation Analysis

Obtaining a correct intent from the Decomposed-FT method requires the two stages in Section 4 to work together effectively. In this section, we investigate error propagation through the two stages using the BiFact decomposition of intents and summaries into atomic facts.

Missed facts, resulting in lowered recall, can occur when a fact is missing in the interaction summarization stage (interaction summarization miss) or the fact can be present in the first stage, but incorrectly dropped in the intent extraction phase (intent extraction miss). An irrelevant or incorrect fact, resulting in lowered precision, can be introduced in the interaction summarization phase and propagated through intent extraction (summarization introduced), or it can be absent from the interaction summarization phase and introduced in the intent extraction phase (intent extraction hallucinated).

Our analysis of the Mind2Web test set is given in Figure 2 using the Decomposed-FT model. The left-hand side, which focuses on recall, shows that the summarization process results in a 16% loss of ground truth facts. Subsequently, intent extraction further reduces the remaining facts by 18%. Effectively, each stage introduces a similar magnitude of error. The right-hand side describes the precision analysis, showing that 8% of the facts predicted by Decomposed-FT were, in fact, hallucinations. This low hallucination rate is attributed to the label processing techniques employed during training. Following that, 20% of the remaining predicted facts were present in the summary but absent from the ground truth, indicating incorrect or irrelevant information in the interaction summarization output and a filtering issue of the intent extractor. We propose this analysis framework to evaluate future two-stage intent extraction methods, aiming to optimize future efforts and assess each stage's impact on overall performance.

7 Computational Cost and Latency Analysis

We conducted a comparative analysis of expected computational cost and latency, for our method vs. the baselines. To provide reliable figures, we relied on published performance benchmarks for the runtime of LLMs, from which expected runtime can be estimated as a function of the number of calls and the number of input and output tokens. We prefer this calculation over directly measuring runtime in our computational environment, since it does not provide easy means to isolate exact

Model	Input tokens	Output tokens	Price per million USD (eq.1)	Latency (eq.2)
E2E	1839	20	191.9	0.24
CoT	1961	127	246.9	0.43
Decomposed FT	2103	622	600	0.6
Decomposed FT (Latency-optimized)	2009	514	406.5	0.24

Table 4: A summary of computational cost and latency

computation times from confounding factors such as network traffic and resource contention. We report computational cost based on pricing of the gemini developer API and expected latency using independent data from https://artificialanalysis.ai (as of June 2025).

Part 1. Estimated total computational cost This is the estimated cost for generating an intent description, summing over all involved LLM calls.

$$\text{Price per million (USD)} = 0.1 \times \text{input tokens} + 0.4 \times \text{output tokens} \quad (1)$$

Part 2. Latency from the last user interaction until the intent description is available for further processing (e.g. for generating follow up suggestions or automation). The latency computation assumes that all screen summaries are generated along the user session, at the end of each interaction step. Thus, end-of-session latency involves only generating the summary of the last screen and the call for generating the intent from all summaries.

We further measure latency for a latency-optimized decomposed variant. In this variant, instead of summarizing the final screen, the visual input is fed directly to the intent generation along with the summaries of all preceding screens, thus incurring latency only for the intent generation LLM call. This variant does not lose quality compared to the original Decomposed FT model.

$$\begin{aligned}\text{Total latency} &\approx \text{Time To First Answer Token} \\ &\quad + \frac{\text{num\_output\_tokens}}{\text{Output Tokens per Second}} \\ &\approx 0.2 + \frac{1}{550} \times \text{num\_output\_tokens} \end{aligned} \quad (2)$$

The size of the input appears to be a negligible factor when the input is between 100-1K tokens [4]. Images are estimated as a fixed 256 tokens each. Overall, the gain in performance from the Decomposed FT approach does come with an additional 2-3x computational cost over the small model baselines that we quantified here. The original Decomposed FT approach would be expected to add latency, which could be an issue in latency sensitive applications, but a minor variant that avoids an extra model call at the end can address that without losing quality.

Comparison to large models Gemini 1.5 pro costs more than 30x per token than Gemini 1.5 Flash 8B - so even if the decomposed FT adds 2-3x cost over the base CoT on a small model, it is still cheaper to run the process on Flash 8B than CoT on Gemini pro. Gemini 2.5 pro is approximately 5x slower per output token and 100x slower in time to first answer token than 2.5 Flash Lite, so the additional latency that comes from using decomposed FT is smaller than the latency that would come from using a large model.

8 Discussion

Our study utilized datasets designed for automation to tackle the challenge of user intent identification, despite their inherent limitations such as noise and information gaps. We observe that fine-tuning alone does not surpass Chain-of-Thought, especially in noisy data scenarios. However, our two stage decomposition exhibited superior performance delivering significant improvements regardless of data quality. This improvement can be attributed to the cleaning process and the combination of prompts and fine-tuning, which effectively mitigated the impact of data noise.

Furthermore, our approach significantly reduced the storage footprint of individual screenshots by summarizing each screen independently, thereby minimizing the required tokens for representation. This reduction in token usage is particularly beneficial for on-device models with limited context windows, enabling them to handle longer trajectories more effectively.

[4] https://artificialanalysis.ai Latency and Output Speed by Input Token Count Context Length

9 Ethical Considerations & Risks

Autonomous agents offer significant innovation, but their development necessitates careful ethical consideration, particularly regarding user privacy. Our research, which aims to interpret user intent from UI interactions, inherently involves sensitive data. We particularly study small models that can run on-device, thereby reducing some of the privacy risks associated with transmitting data to external servers. Furthermore, accurately understanding user intents can greatly benefit users through enhanced personalization, improved work efficiency, and facilitating future recall of past activities on their devices. While this work focuses on intent understanding, the development of agents capable of autonomously completing actions requires extreme care. The potential for for misalignment with user intentions and the need for robust safeguards must be thoroughly addressed to ensure responsible deployment.

10 Limitations

We acknowledge several discrepancies between our datasets and real-world user behavior. The datasets predominantly feature English-language, U.S.-centric web interactions, restricting our analysis to this specific demographic. In contrast, real-world users frequently navigate multiple applications, adapt their goals on the fly, and exhibit varying levels of digital literacy, resulting in more complex and unpredictable interaction patterns. The Mind2Web dataset's single-website limitation further deviates from the multi-site nature of typical user tasks. Additionally, our study's reliance on Android and web environments limits the generalizability of our findings to other platforms.

References

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, et al. 2021. Uibert: Learning generic multimodal representations for ui understanding. *arXiv preprint arXiv:2107.13731*.

Omri Berkovitch, Sapir Caduri, Noam Kahlon, Anatoly Efros, Avi Caciularu, and Ido Dagan. 2025. Identifying user goals from ui trajectories. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 2381–2390.

Andrea Burns, Deniz Arsan, Sanjna Agrawal, Ranjitha Kumar, Kate Saenko, and Bryan A Plummer. 2022. A dataset for interactive vision-language navigation with unknown command feasibility. In *European Conference on Computer Vision*, pages 312–328. Springer.

Sapir Caduri, Anatoly Efros, Noam Kahlon, Danielle Cohen, Yoni Halpern, and Ido Dagan. 2025. Bi-fact: A bidirectional factorization-based evaluation of intent extraction from ui trajectories. *arXiv preprint arXiv:2502.13149*.

Dingxin Cheng, Mingda Li, Jingyu Liu, Yongxin Guo, Bin Jiang, Qingbin Liu, Xi Chen, and Bo Zhao. 2024. Enhancing long video understanding via hierarchical event-based memory. *CoRR*.

Janara Christensen, Stephen Soderland, Gagan Bansal, et al. 2014. Hierarchical summarization: Scaling up multi-document summarization. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 902–912.

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.

Yicheng Fu, Raviteja Anantha, Prabal Vashisht, Jianpeng Cheng, and Etai Littwin. 2025. Ui-jepa: Towards active perception of user intent through onscreen user activity. In *User Modeling, Adaptation and Personalization, UMAP 2025 (to appear)*.

Gemini Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating factual consistency evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3905–3920, Seattle, United States. Association for Computational Linguistics.

A Jiménez-Ramírez. 2024. A screenshot-based task mining framework for disclosing the drivers behind variable human actions. *Information Systems*, 121:102340.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Daniel Li, Thomas Chen, Albert Tung, and Lydia B Chilton. 2021a. Hierarchical summarization for longform spoken dialog. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 582–597.

Gang Li and Yang Li. 2022. Spotlight: Mobile ui understanding using vision-language models with a focus. *arXiv preprint arXiv:2209.14927*.

Toby Jia-Jun Li, Lindsay Popowski, Tom Mitchell, and Brad A Myers. 2021b. Screen2vec: Semantic embedding of gui screens and gui components. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024. On the effects of data scale on computer control agents. *arXiv preprint arXiv:2406.03679*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems*, 36:59708–59728.

Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. 2021. Screen2words: Automatic mobile ui summarization with multimodal learning. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 498–510.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024a. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *CoRR*.

Shuai Wang, Weiwen Liu, Jingxuan Chen, Weinan Gan, Xingshan Zeng, Shuai Yu, Xinlong Hao, Kun Shao, Yasheng Wang, and Ruiming Tang. 2024b. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*.

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.

Jiajun Xu, Zhiyuan Li, Wei Chen, Qun Wang, Xin Gao, Qi Cai, and Ziyuan Ling. 2024. On-device language models: A comprehensive review. *arXiv preprint arXiv:2409.00088*.

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

Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. 2025. Aria-ui: Visual grounding for gui instructions. In *ICLR 2025 Workshop on Foundation Models in the Wild*.

Guanhua Zhang, Mohamed Adel Naguib Ahmed, Zhiming Hu, and Andreas Bulling. 2025. Summact: Uncovering user intentions through interactive behaviour summarisation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Bin Zhao, Maoguo Gong, and Xuelong Li. 2022. Hierarchical multimodal transformer to summarize videos. *Neurocomputing*, 468:360–369.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v (ision) is a generalist web agent, if grounded. In *Forty-first International Conference on Machine Learning*.

A Preprocessing Details

The data preprocessing pipeline for the Mind2Web and AndroidControl datasets involved several tailored steps. For image preprocessing, we adopted a holistic approach, where entire screenshots were resized as necessary to conform to model input specifications, deliberately avoiding patch-based methods. For the Mind2Web dataset, full-webpage screenshots were first processed by cropping them to a uniform size of 1280×768. This crop was specifically defined by the bounding box of the user's interaction, ensuring this critical area dictating the action was captured within random margins before the image was resized. In contrast, the AndroidControl dataset, with its uniformly sized mobile screenshots 1080×2400, only necessitated resizing and the subsequent visual overlaying of a bounding box derived from the available user action coordinates. The action extraction process also varied: Mind2Web provided action details directly (which includes information like bounding box coordinates of the target element), whereas for AndroidControl, it was necessary to identify the interacted UI element using its coordinates (which define its bounding box) and then retrieve its name via the accessibility tree. Note that for Gemini experiments on Mind2Web, we additionally filtered examples by domain name to comply with Google-extended policy[5]. The resulting test set size was

[5]https://blog.google/technology/ai/an-update-on-web-publisher-controls/

reduced from 1005 to 681.

For the gold standard extracted goal, we use the high-level goal for each dataset. As mentioned in Section 2.2, the annotation process of AndroidControl was less rigorous than that of Mind2Web, resulting in noisier labels. Furthermore, AndroidControl labels, designed to simulate real user instructions, often contain irrelevant information that cannot be inferred from the trajectory (e.g., “I’m hungry, order an olive pizza from DoorDash”). To mitigate the impact of this noise, we cleaned the labels using Gemini 1.5 Pro (Prompt in I2). This cleaning still doesn’t completely provide clean goals like Mind2Web’s validation process. We find that even after applying a prompt-based cleaning, manual validation on 100 examples (following the annotation protocol in Berkovitch et al. (2025)) makes changes to $\sim 30\%$ of the label intents.

Finally, a common preprocessing step was applied to the goals from both Mind2Web and AndroidControl. We noted that the specific application or website name was often available with the interaction data. Yet, this platform-specific information is not directly or consistently inferable from the visual input of screenshots and the action sequences alone. To address this, we programmatically isolated these platform identifiers from the core user intent, restructuring the label into an "appname/website; intent" format. This approach serves a dual purpose: it retains platform information, useful for contextual fine-tuning, and also allows for the simple removal of this identifier prior to evaluation. Such removal ensures that our assessment accurately reflects the model’s capability to interpret user intent, rather than its ability to identify the specific platform. As a result, any distorting effects from platform recognition on the evaluation metrics are prevented, which is important given that platform identification is not a primary focus of this study.

B Fine-Tuning Details

For the fine-tuning process, we adapted slightly distinct approaches for the Gemini and Qwen models, largely adhering to established practices.

The Gemini models were fine-tuned following procedures analogous to those described described at `https://ai.google.dev/gemini-api/docs/model-tuning`. A learning rate of $1e-6$ was used without specific hyperparameter tuning, and a batch size of 16 was employed. Training proceeded for a maximum of two epochs, with checkpoints saved at intervals of 20 steps. The Gemini model chosen was the one that achieved the minimum negative log-likelihood on its respective validation data, effectively employing an early stopping strategy based on this metric. Similarly, for the Qwen2-VL-7B model, we followed methodology outlined in the Hugging Face VL fine-tuning cookbook[6]. This included adopting the author’s recommended hyper-parameters, such as the default learning rate of $2e-4$. Due to memory constraints, a batch size of 1 was employed for Qwen. Training was also conducted for a maximum of two epochs, and checkpoints were saved every 20 steps, as suggested in the tutorial. Consistent with the Gemini models, the final Qwen model was selected to minimize negative log-likelihood on validation data. For the AndroidControl dataset, we used 5,000 training examples and 137 validation examples randomly sampled from the train set. For Mind2Web we used 900 training examples and 90 validation examples.

C Expert Annotation Labeling

Table 5 expands on the results shown in Table 2 of the main text, providing the detailed BiFact precision and recall scores in addition to the F1 score for the comparison against expert annotations and original dataset labels on the AndroidControl dataset. As is evident from these numbers, the increase in recall when evaluating against expert labels is particularly significant. This suggests that many of the facts included in the original dataset labels were not actually fulfilled (or were unfulfillable) within the recorded user interaction trajectories. Consequently, the higher recall achieved against the expert-annotated labels more accurately reflects the model’s performance on verifiable and achievable intents.

D Human Preference Annotation

We presented the rater with a full trajectory of screenshots and actions, and then asked the following question: “After you have seen the trajectory, which intent better describes the trajectory? A or B.” The choices A and B contained either CoT or Decomposed-FT. The order of the two options were randomized in each question and the names of the methods were not shown to the respondent. The

[6]`https://huggingface.co/learn/cookbook/en/fine_tuning_vlm_trl`

	Expert Labels			Dataset Labels		
	F1	Precision	Recall	F1	Precision	Recall
Gemini Flash 8B						
CoT	0.652	0.674	0.714	0.580	0.600	0.663
E2E-FT	0.590	0.636	0.623	0.565	0.626	0.601
Decomposed-FT	0.701	0.714	0.762	0.596	0.639	0.655
Gemini-1.5-Pro						
CoT	0.724	0.688	0.862	0.635	0.617	0.746

Table 5: A comparison of BiFact F1, precision and recall scores for intent prediction on the AndroidControl dataset, using expert annotations and dataset labels as ground truth.

	Mind2Web				**AndroidControl**			
		BiFact		Bi-NLI		BiFact		Bi-NLI
Method	F1	Precision	Recall		F1	Precision	Recall	
Gemini Flash 8B								
CoT	0.660	**0.751**	0.656	0.326	**0.594**	**0.628**	0.660	0.302
Decomposed-non-FT	**0.718**	0.717	**0.792**	0.221	0.528	0.488	**0.719**	0.185
Gemini-1.5-Pro								
CoT	0.721	**0.761**	0.740	0.331	**0.634**	**0.612**	0.767	0.347
Decomposed-non-FT	**0.732**	0.700	**0.859**	0.213	0.512	0.441	**0.791**	0.228

Table 6: BiFact results on the Mind2Web and AndroidControl datasets using Gemini 1.5 Flash 8B and Gemini 1.5 Pro.

	DOMAIN UNSEEN			TASK UNSEEN			WEBSITE UNSEEN		
	F1	Prec.	Recall	F1	Prec.	Recall	F1	Prec.	Recall
Gemini Flash 8B									
CoT	0.656	0.767	0.641	0.651	0.717	0.655	0.692	0.786	0.664
E2E-FT	0.665	0.686	0.686	0.606	0.631	0.618	0.674	0.618	0.694
Decomposed-FT	**0.747**	**0.800**	**0.752**	**0.732**	**0.823**	**0.710**	**0.817**	**0.899**	**0.785**
Gemini-1.5-Pro									
CoT	0.723	0.774	0.734	0.731	0.762	0.762	0.756	0.785	0.774

Table 7: Detailed BiFact-based performance breakdown on different subsets of the Mind2Web test set.

decoding of choices to model name was only done after the rater had finished the task.

E Results Using an Alternative LLM for Evaluation

To demonstrate that our findings are robust to the choice of the underlying model for our evaluation metric, we present a replication of our main results in Table 8. In this analysis, we replaced the LLM used by BiFact, substituting Gemini Pro with Gemini Flash.

The results show that the overall performance trends and the relative ranking of the evaluated models remain consistent, confirming that our main conclusions are not dependent on a specific evaluation LLM.

F Detailed Test Set Performance Breakdown

The test-sets for Mind2Web (Deng et al., 2024) and AndroidControl (Li et al., 2024) have multiple types of unseen data. In this appendix, we provide a more detailed breakdown of the BiFact performance scores on each of the subsets of the test sets.

The detailed performance breakdown on the Mind2Web test set, as presented in Table 7, offers several key insights into model generalization. As might be expected, the standard end-to-end fine-tuned (E2E-FT) model using Gemini Flash 8B performs worse than the COT approach on data from previously unseen tasks (TASK UNSEEN) and unseen websites (WEBSITE UNSEEN). However, its performance is notably on par with the COT model in the DOMAIN UNSEEN category. This pattern suggests that while the E2E-FT model may be tuned somewhat towards specific tasks and characteristics of websites present in its training data, its ability to handle completely new types of tasks at a broader domain level is not further compromised compared to the prompt-based COT method. In stark contrast, the Decomposed-FT model (Gemini Flash 8B) demonstrates strong performance, consistently outperforming both the COT and E2E-FT methods across all three challenging unseen categories (DOMAIN, TASK, and WEBSITE UNSEEN). Furthermore, its performance in these generalization scenarios surpasses that of the larger Gemini 1.5 Pro (COT) model, particularly on unseen domains and websites. This robust performance can be attributed to the sophisticated fine-tuning scheme employed for the Decomposed-FT model. This comprehensive approach—which involves using prompts for structured interaction summarization, meticulous data cleaning through label refinement, and then fine-tuning on these processed, higher-quality inputs—makes the model significantly less vulnerable to common problems associated with regular fine-tuning, such as overfitting to training set specifics or sensitivity to label noise. The primary limitation highlighted by Table 7, when comparing the Decomposed-FT model (using Gemini Flash 8B) to the Gemini-1.5-Pro model, is its slightly lower recall in the TASK UNSEEN category (0.710 for Decomposed-FT vs. 0.762 for Pro). This specific gap suggests that while highly effective, the training scheme could potentially benefit from exposure to a more diverse range of task examples to further enhance generalization for entirely novel tasks, even when encountered within familiar website or domain contexts.

G Detailed Analysis of the non-finetuned Decomposed ablation

To provide a complete picture and address potential inquiries regarding the performance of our approach without the crucial fine-tuning of the intent extraction stage, this section offers a more in-depth analysis of the non-finetuned ablation of our decomposed method. Table 6 presents a comparison of the non-finetuned decomposed ablation with the CoT baseline. Since neither method requires fine-tuning, we can demonstrate their performance across both small and large models. Notably, our full method, Decomposed-FT, surpasses both the non-finetuned decomposed variant, as evidenced in the ablation study in Table 3, and the CoT baseline, as shown in Table 1. The results in Table 6 indicate that the non-finetuned decomposed method demonstrates strong performance on the Mind2Web dataset, yet underperforms considerably compared to the CoT baseline on the Android Control dataset, a trend consistent across both small and large model sizes. This performance differential can be attributed to the inherent verbosity of the non-finetuned decomposed method, which generates a higher average number of atomic facts per predicted intent compared to CoT. Specifically, for Android Control, the non-finetuned decomposed approach produced an average of 4.0 facts versus 2.5 for CoT, while gold has 3.0 facts. For Mind2Web, these figures were 4.4 for Decomposed versus 2.8 for CoT, while

	Mind2Web			AndroidControl		
		BiFact			BiFact	
Method	F1	Precision	Recall	F1	Precision	Recall
Gemini Flash 8B						
CoT	0.681	0.796	0.656	0.623	0.675	0.681
E2E-FT	0.678	0.726	0.680	0.651	0.72	0.679
Decomposed-FT	**0.794**	**0.874**	**0.771**	**0.685**	**0.741**	**0.718**
Gemini-1.5-Pro						
CoT	0.746	0.805	0.757	0.701	0.695	0.811

Table 8: Replication of Main Results (Table 1) Using an Alternate LLM. BiFact results on the Mind2Web and AndroidControl datasets, using Gemini-Flash as the underlying LLM for BiFact rather than Gemini-Pro. This demonstrates that our main findings are robust to the choice of the underlying LLM used for evaluation.

the gold has 4.4 facts. This increased verbosity correlates with the observed lower precision of the non-finetuned decomposed method across both datasets. Conversely, it also correlates with the superior fact-level performance (BiFact), of the decomposed method on Mind2Web, where the gold annotations are themselves more verbose than those for Android Control. As our main results show, subsequent fine-tuning of the second-stage model demonstrably improves precision by training the model to selectively include only the most relevant facts in the final intent formulation, leading to the significantly better performance of our proposed method.

H Comparison with SummAct

In SummAct (Zhang et al., 2025), each input interaction is represented as a short textual description of the specific UI element with which the user interacted and the respective user action. Intent extraction is then performed hierarchically, in two steps. First, the sequence of interactions is summarized into a shorter sequence of mid-level sub-goals, using few-shot prompting. Then, the sequence of sub-goals is further summarized into the final high-level intent description, using a model that is fine-tuned to produce the gold intent given the output of the first step. While this method seems somewhat similar to ours in that it decomposes intent extraction into two subsequent steps, the two methods differ substantially, in both their input and their decomposed steps. SummAct considers only localized textual input, describing just the UI element with which the user interacted, while ignoring the full screen and its visual layout. This restricts the model from understanding the wider context, like the elements the user didn't choose to interact with, or visual cues that may influence user behavior. In contrast, our model, considers the full screenshot information. Thus, in our first step, the model generates a textual summary for each interaction step, which considers the broader screen context. Subsequently, our second step directly summarizes these interaction level descriptions into the final high-level intent description, using a sophisticated fine-tuning approach that synchronizes the inputs for this step with the gold output. This method does not require an intermediate step of generating sub-goal descriptions, like SummAct. Further, while SummAct's finetuned model required intervention with the attention mechanism to perform well, which may not be accessible or practical in various settings, our finetuning approach allowed us to finetune available models as is, without further intervention. A potential advantage of SummAct's text-only approach may be computational efficiency as it consumes smaller inputs and uses fewer model calls.

I Prompts

```
You are analyzing a user's session on a mobile app. Each session consists of a sequence of
screenshots and actions that will appear at the end of this prompt. Your goal is to understand
the user's overall intent based on the series of interactions provided.

Instructions:

1. Analyze Each Step in the Sequence**:
   - What is displayed in the screenshot?
   - What action did the user take?
   - Why might the user have taken this action?
   - What specific details are relevant? (e.g., dates, items, locations, quantities)

2. Summarize the User's Goal:
   - After analyzing each screenshot-action pair, combine insights to determine the user's
overall objective.
   - Include all observed details to make the goal clear and specific.

Output:

1. **Reasoning**:
   - Provide a step-by-step analysis of each screenshot and action pair.
   - Focus on the user's likely intentions and relevant details observed in the input sequence.

2. **Final Answer**:
   - Summarize the user's overall goal in one concise sentence.
   - Start with an action verb. Phrase the action in the imperative form.
   - Include all specific details observed in the input sequence.
   - Avoid using any phrases or structures from the instructions or examples above.
   - Your final answer should start with the appname, followed by a semicolon, and then the
inferred goal (example: "eBay; order a basket ball").

Important Notes:

- Do not reuse or paraphrase any examples provided in the instructions.
- Base your response solely on the screenshots and actions in the input sequence.
- Each session is unique ensure your final answer reflects the specific details of this session
alone.

The format of the the output should json:

{{
    "reasoning": "your reasoning here",
    "final_answer": "your final answer here"
}}

---

The sequence of screenshots and actions for this session will now follow:
```

Figure I1: CoT model prompt, used as the baseline as described in Section 3

Your task is to rephrase instructions given by users to automated agents that execute tasks on the user's phone.
Rephrase the instruction in the imperative mood, starting with a verb, and remove any text irrelevant to the user's objective.
Add the app name before the instruction, in the following format: "App name; Instruction".
If the app is now known, use "Unknown app; Instruction".
Correct any spelling or punctuation errors as needed.

Here are a few examples of rephrased instructions:
Input: I am tired of the hustle and bustle of the world. I want to just have a peaceful mind. Play the classic song "Casta diva by Maria Callas" in the Dailymotion app
Output: Dailymotion; Play the song "Casta diva by Maria Callas"

Input: I want to write the review comment, Perfect! My favorite dessert for this recipe
Output: Unknown app; Write the review comment: "Perfect! My favorite dessert for this recipe."

Input: Open TickTick app and share the wedding plan task on dwbscratchid3@google.com through Gmail
Output: TickTick; Share the wedding plan task on dwbscratchid3@google.com through Gmail

Input: I'd like to forward Google Community team emails to Cerebra Research at dbwscratch.test.id4@gmail.com.
Output: Gmail; Forward Google Community team emails to Cerebra Research at dbwscratch.test.id4@gmail.com.

Input: I am looking for a rental place in St. John, USA, under $4,000, so search for rental properties for me in St. John on the Redfin app.
Output: Redfin; Search for rental properties in St. John, USA under $4,000.

Your test instruction:

Figure I2: AndroidControl cleaning prompt, used to automatically clean the dataset as described in Section 5.2

```
You are evaluating user behavior within a mobile app.  Given a screenshot of the app interface
and a description of the user's action, your task is to provide a comprehensive summary of the
user's intent and the specifics of their interaction.

**Instructions:**

1. **Analyze the Input:**
   - Carefully examine the provided screenshot.
   - Interpret the user's action, including any additional information provided.

2. **Extract Key Information:**
   - Identify all relevant elements on the screen (e.g., buttons, text fields, images).
   - Pinpoint the user's specific action (e.g., tap, scroll, input text).
   - Note important details like dates, times, locations, quantities, or text content.

3. **Format the Output:**
   - **Output a newline-delimited list where each item represents a distinct piece of information.
   - **Do not include any explanatory text or labels.** Just the newline-delimited list.
   - Example:
      User viewed product details for iPhone 14 Pro Max.
      User added the product to their shopping cart.
      User selected the '256GB' storage option.

**Input:**

- **Screenshot:** <img>
- **Action:** {{action}}

**Note:**

- The action description may include contextual information like text content, direction (e.g.,
'swiped left'), app name, or UI element name.
- The screenshot may contain a red bounding box highlighting the interacted element.
```

Figure I3: Interaction summarization prompt, used to summarize single screen interaction as explained Section 4.1

```
You are given a summarized user journey, consisting of screen summaries that describe what
the user saw on each screen and what they did. Your task is to analyze this journey and infer
the user's intent.

Your output should be a concise description of the user's intent that includes:

1. **The user's primary goal:** What were they ultimately trying to achieve?
2. **All apps involved:** List every app used in the journey.
3. **Key actions:**  Highlight specific actions within the summaries that reveal the intent (e.g.,
search queries, filter selections,  options chosen). Avoid reporting purely navigational actions.

**Important Considerations:**

* **Complex Intents:** Longer journeys may involve evolving or multiple intents.  Strive to
identify the most plausible explanation for the user's actions, even if their initial goal shifted.
* **Conciseness:** Aim for 2-3 sentences that capture the essence of the intent.
* **Output Format:**  "AppName; Intent description" (e.g., "Amazon; User viewed the product
page for 'noise-canceling headphones',  added them to their cart, and proceeded to checkout.")

Your response should contain nothing but the output in the specified format. Do not add any
additional text or explanations.

**Important: ALL information should be extracted from the summaries. Do NOT introduce any
new information.**

**Output examples:**

Expedia; User launched the Expedia app, searched for flights from Paris (CDG) to London
(LHR) departing on January 7th, filtered results by "non-stop flights" and "lowest price", and
finally selected a British Airways flight departing at 10 PM.

Clock; User opened the Clock app, tapped on the "Alarm" tab, set a new alarm for 7:00 PM
tomorrow, toggled the "Snooze" option off, and saved the alarm.

Spotify; User opened Spotify, searched for "holiday music", tapped on the "Create Playlist"
button, named the playlist "Christmas 2024", and added songs like "Jingle Bells" and "Silent
Night" to the playlist.

Gmail; User opened the Gmail app, opened an email from "Bank of America" with the subject
"Your November Statement", tapped on the link to view the PDF statement, then navigated back
to their inbox and replied to an email from their boss with the subject "Project Update."
```

Figure I4: Session-level intent prompt, used to fuse single interaction summarise to a single intent as explained in Section 4.2

```
You are given a summary of a user trajectory and an inferred goals of the user.

Your task is to rewrite the inferred goal in a way that it only contains information that is
present in the summaries.
Any information that is not present in the summaries should be removed.

Summaries: {{[combined summaries]}}
Inferred goal: {{[clean goal]}}

The output format should be a json object with the following format:
{{{{
    "facts_in_summaries": ["fact1", "fact2", ...],
    "facts_not_in_summaries": ["fact3", "fact4", ...],
    "rewritten_goal": "the rewritten goal in plain text"
}}}}
```

Figure I5: Label refinement prompt, used to refine the label prior to the fine-tunning step as explained in Section 4.2