# Predicting Web-Agent Failure Risk from Trajectory Prefixes

Anonymous ACL submission

## Abstract

Reliable web agents are critical for deployment in real-world interactive environments. Unlike single-turn language generation, web-agent decision making is inherently sequential and context-dependent, so reliable deployment requires predicting failure risk from the evolving trajectory prefix. In this work, we study prefix-level failure-risk prediction for web agents: estimating, whether an ongoing trajectory will eventually fail. Based on our analysis on agentic failure trajectories, we propose an interpretable Macro–Micro risk representation and train a lightweight calibrated predictor over trajectory prefixes. Experiments on WebArena-Lite and Online Mind2Web show that our method improves selective risk ranking and failure discrimination over trajectory-calibration baselines, especially on E-AURC and AUROC. Further analyses show that the learned risk signals support early failure detection under controlled false-cut budgets and transfer across website regimes. These results demonstrate the effectiveness and practical utility of our Macro–Micro framework for reliable web-agent failure-risk prediction.

## 1 Introduction

Reliable web agents require the ability to detect failure risk before a task terminates. Web agents automate complex tasks on real-world websites by interacting with dynamic environments through sequential perception, reasoning, and action. In this setting, a single erroneous or overconfident decision can propagate through later interactions and compound into eventual task failure (Yao et al., 2022; Oh et al., 2026). Thus, beyond estimating final-task confidence, web agents require prefix-level risk prediction: given an intermediate execution history, estimating whether the trajectory is likely to fail before the final outcome is observed.

Uncertainty quantification (UQ) provides a natural lens for prefix-level risk prediction, since failures can arise from heterogeneous uncertainty sources across the trajectory. By observing 3,364 web-agent trajectories in our preliminary study, we identify two recurring failure patterns. At the trajectory level, agents may repeat ineffective actions, enter behavioral loops, or accumulate execution errors. At the decision level, they may be uncertain about their intention, grounding target, anticipated interface change, or action value. Recent agentic UQ work has begun to address multi-step settings through trajectory-level calibration and uncertainty propagation modeling (Duan et al., 2025; Zhang et al., 2026a; Oh et al., 2026). However, these methods are either not tailored to web agents or provide coarse trajectory-level estimates without interpretable uncertainty signals at each execution prefix.

In this work, we study prefix-based risk prediction for web agents. Given the execution history up to step $k$, our goal is to estimate the probability that the trajectory will eventually fail before the final outcome is observed. We propose an uncertainty-informed macro–micro framework for prefix-level risk prediction. The macro component captures trajectory-level behavioral risk signals, such as repetition, loops, and execution errors, while the micro component captures latent decision uncertainty over intention, grounding target, anticipated interface change, and action value. These features are extracted from each execution prefix and fed into a lightweight predictor to estimate future failure risk. This enables early risk detection and provides interpretable links between local decision instability and eventual trajectory failure.

We evaluate the proposed framework on two web-agent benchmarks, WebArena-Lite (Liu et al., 2024) and Online Mind2Web (Gou et al., 2025), across five backbone models. Results show that combining macro and micro signals consistently improves risk prediction over representative baselines in both risk-ranking (AURC) and failure-

discrimination (AUROC) metrics. In summary, our contributions are:

1. We propose complementary macro and micro interpretable features for web-agent risk prediction, capturing both environment-interaction patterns and latent decision-making uncertainty.

2. We introduce a **prefix-based risk prediction** framework that estimates eventual task-failure risk from the execution history up to step $k$, enabling early and actionable risk assessment during agent execution.

3. We evaluate our framework on two web-agent benchmarks and five backbone models, showing consistent gains over representative uncertainty and confidence baselines in AURC, AUROC, and Brier score, together with interpretability analyses and evidence of cross-model and cross-task generalization.

## 2 Related Work

### 2.1 Uncertainty Quantification for LLMs

UQ methods for LLMs are commonly divided into white-box and black-box approaches. White-box methods derive uncertainty from internal signals such as token logits (entropy, perplexity) (Manakul et al., 2023; Bakman et al., 2024; Malinin and Gales, 2020), optionally aggregated over reasoning traces (Da et al., 2025; Li et al., 2025; Zhang and Zhang, 2025). Black-box methods rely on output behavior, such as consistency under multi-sampling or semantically equivalent paraphrases (Kuhn et al., 2023; Kossen et al., 2024; Qiu and Miikkulainen, 2024), or prompting the model to verbalize its confidence (Kadavath et al., 2022; Tian et al., 2023; Tao et al., 2024). A common limitation is that these methods are designed for single-turn outputs, failing to model how confidence evolves across a sequence of interdependent decisions.

### 2.2 Uncertainty-informed risk prediction for web agents.

Recent work has shown that uncertainty in LLM agents can propagate through sequential decisions and compound into task failure (Zhang et al., 2026b; Duan et al., 2025; Oh et al., 2026). Several studies further explore uncertainty or confidence signals in web agents, using them for adaptive planning, MCTS-based reasoning, test-time scaling, or retry decisions (Zhang et al., 2026c; Lee et al., 2026; Ou et al., 2025). However, these methods primarily use uncertainty as an online control signal rather than estimating calibrated failure risk from intermediate execution prefixes. Post-hoc agent UQ methods study uncertainty propagation or trajectory-level calibration (Duan et al., 2025; Zhang et al., 2026a), but they either provide coarse scalar uncertainty estimates or require white-box token probabilities. In contrast, our work targets black-box web agents and predicts prefix-level task-failure risk using interpretable macro-level behavioral signals and micro-level latent decision uncertainty.

## 3 Methodology

### 3.1 Problem Formulation

We formalize web-agent interaction as a sequential decision process and define the prefix-level risk prediction problem.

A web agent interacts with a dynamic browser environment over multiple steps to fulfill a user task $q$. At each step $t$, the agent receives a page-state observation $O_t$ and produces a structured action decision $D_t = (R_t, A_t)$, where $R_t$ is the textual reasoning trace and $A_t$ is the executable browser action. Let

$$H_t = (O_0, D_0, O_1, D_1, \ldots, O_{t-1}, D_{t-1}) \quad (1)$$

denote the interaction history before step $t$. The full *context* available to the agent at step $t$ is $X_t = (q, O_t, H_t)$. The agent's behavior is then governed by a context-conditional decision policy, $D_t \sim \pi(\cdot \mid X_t)$. Each decision changes the environment and thereby shapes all future contexts.

We study *prefix-level risk prediction*: given the trajectory prefix

$$\tau_{\leq t} = (q, O_0, D_0, \ldots, O_t), \quad (2)$$

estimate the probability of eventual failure, $r_t = P(\text{failure} \mid \tau_{\leq t})$, using only information available up to step $t$. This *prefix validity* constraint ensures $r_t$ can be deployed online for early intervention and user-facing confidence reporting.

To ground this formulation, we next examine *how* web agents fail in practice and *why* existing UQ methods cannot capture those failures.

### 3.2 Empirical Observations from Error Annotation

To systematically analyze agent failures, we collect trajectories from WebArena-Lite and Online

Mind2Web, annotating 3,364 failure steps using a structured taxonomy adapted from AgentRx (Barke et al., 2026a). These LLM-based annotations are verified against a human-validated subset (see Appendix B). As detailed in Table 3, the empirical distribution reveals that over half of all failures are driven by two distinct structural patterns: *error behavioral patterns* (∼23%) and *intention–action–outcome mismatches* (∼31%).

**Observation 1: Error Behavioral Patterns.** Approximately 23% of failures involve repetitive error patterns, such as re-executing a previously failed action or repeatedly backtracking along the same wrong path. These failures are difficult for standard UQ signals because each individual step can appear locally reasonable and confident. The risk is instead expressed across the prefix—through repetition, loops, and failure to recover—so it is largely invisible to methods that inspect only the current action.

**Observation 2: Intention–Action–Expectation Mismatch.** Roughly 31% of failures reflect internal mismatch within a step. By examining $(O_{t-1}, D_t, O_t)$, we observe two common cases: the agent forms a plausible intention but executes the wrong action, or it executes the intended action while expecting an incorrect environment response. Since existing UQ methods usually treat $X_t \to D_t$ as an opaque action choice, they score only the final decision and miss these alignment gaps inside the decision process.

Together, these observations establish a dual requirement for robust web-agent risk prediction: models must look *across* steps to detect behavioral loops, and look *inside* each step to expose decision misalignment. This directly motivates our feature design in Section 3.3.

### 3.3 Macro-Micro Feature Design

The two observations in Section 3.2 point to structurally different failure sources and therefore demand structurally different features. Observation 1 requires features that look *across* steps to detect agent-environment interaction patterns; Observation 2 requires features that look *inside* each step to expose decision misalignment. We design **Macro** features for the former and **Micro** features for the latter.

**Macro Features: Trajectory-Level Behavioral Patterns** Observation 1 shows that behavioral loops can trap an agent across multiple steps while each individual step appears normal. Detecting such patterns requires examining the sequence of executed actions and the agent–environment interaction history across the prefix. We therefore construct Macro features as interpretable trajectory-level signals computed from the observed decisions $D_t = (R_t, A_t)$, execution metadata as well as the reasoning reflection, without additional model inference. They cover three main groups, yielding $\mathbf{x}_t^{\text{Macro}} \in \mathbb{R}^{31}$(full definitions in Appendix D.):

**Action repetition and loops.** These features capture short cyclic behaviors and stalled exploration, including consecutive identical action pairs, alternating two-action cycles, loop suffix status, distinct action count, repetition ratio, and maximum action frequency.

**Behavioral and environmental errors.** These features summarize whether the prefix shows abnormal interaction dynamics or execution failures, including action-type counts (click, type, scroll, etc.), grounding errors, parsing errors, and consecutive error streaks.

**Inner thinking length.** Reasoning length reflects how much deliberation the agent needs before acting; unusually long or increasing traces can indicate uncertainty, recovery attempts, or confusion. We track the current, maximum, and mean reasoning length over the prefix, together with keyword indicators for self-reported ineffective actions, misplaced context, and environment errors in the reasoning trace.

**Micro Features: Step-Level Latent Decision Decomposition**

Observation 2 reveals that $X_t \to D_t$ is not a monolithic mapping. As shown by the intention–action and action–outcome mismatches in Section 3.2, an agent decision involves three distinct stages: deciding what to achieve, selecting which action to execute, and anticipating what will happen afterward. We formalize these as three latent variables at each step: the agent's **intention** $I_t$ (the current subgoal), the **decision** $D_t$ (the action to execute), and the anticipated **state change** $C_t$ (the agent's expectation about how the interface will change after executing $D_t$). These describe a structured decision process:

$$X_t \to I_t,\ (X_t, I_t) \to D_t,\ (X_t, I_t, D_t) \to C_t, \tag{3}$$

which separates three sources of uncertainty: **intention uncertainty**, **grounding uncertainty**, and

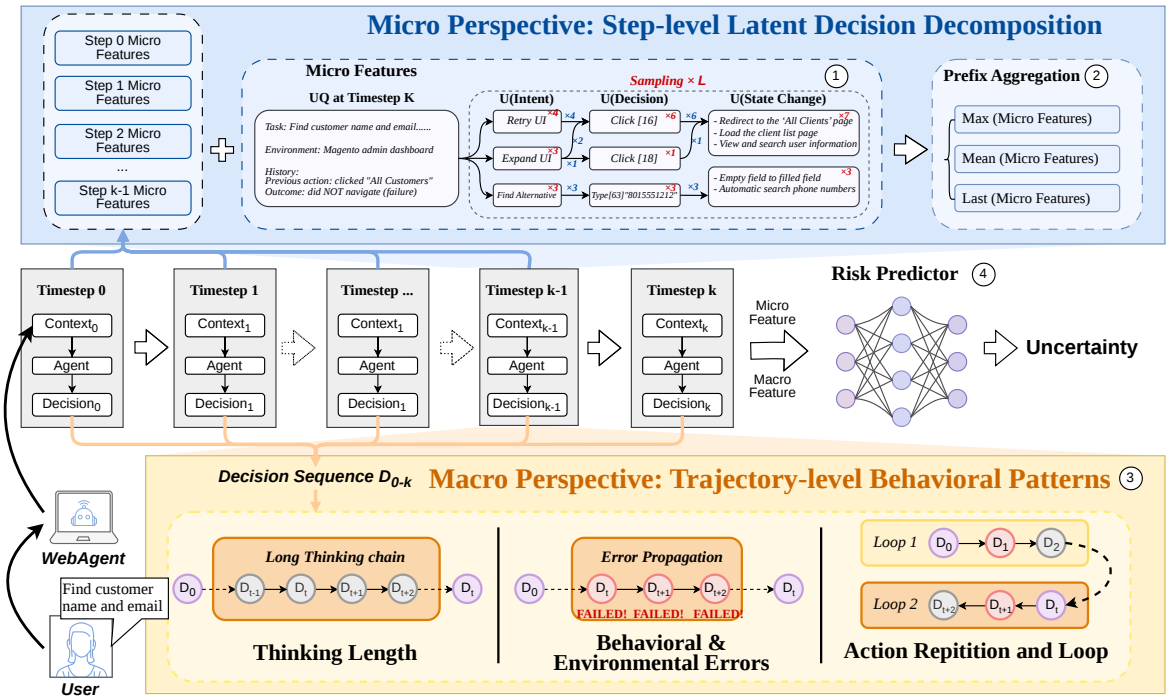


Figure 1: Overview of the proposed UQ framework: Micro features decompose each step decision into intention, decision, and state-change components, then aggregate over the prefix; Macro features capture trajectory-level behavioral anomalies such as loops, repetition, and error accumulation. Both feature families feed into a calibrated risk predictor.

**state-change uncertainty**.

**Estimating the latent distribution.** To quantify these uncertainties, we estimate the latent distribution at each step $s \leq t$ through structured multi-sampling. We fix the context $X_s$, query the agent $N$ times, and obtain $N$ tuples $\big(\tilde{I}_s^{(m)}, D_s^{(m)}\big)_{m=1}^{N}, \tilde{C}_s^{(m)}$, where $\tilde{I}_s^{(m)}$ and $\tilde{C}_s^{(m)}$ are free-form text and $D_s^{(m)}$ is the structured action. From the clustered samples, we estimate the empirical joint distribution $\hat{p}_s(i, c, d \mid X_s) = \frac{1}{N}\sum_{m=1}^{N} \mathbf{1}[I_s^{(m)} = i,\, C_s^{(m)} = c,\, D_s^{(m)} = d]$ and factorize it via the chain rule:

$$p(I_s, D_s, C_s \mid X_s) = p(I_s \mid X_s)\, p(D_s \mid I_s, X_s)\, p(C_s \mid D_s, I_s, X_s) \tag{4}$$

This factorization isolates intention, grounding, and state-anticipation uncertainty within one probabilistic representation. **Feature extraction and prefix aggregation.** From this factorized distribution we derive six scalar indicators: the marginal entropies $H(I_s)$, $H(D_s)$, $H(C_s)$; the conditional entropies $H(D_s \mid I_s)$ and $H(C_s \mid D_s, I_s)$; and the intention–decision mutual information $C_I = H(D_s) - H(D_s \mid I_s)$. Full definitions are provided in Appendix D. To obtain prefix-level features at step $t$, we aggregate each indicator over all steps 0 to $t$ using three operators (mean, max, value at $t$), yielding $\mathbf{x}_t^{\text{micro}} \in \mathbb{R}^{18}$. Since intentions and anticipated state changes are expressed in natural language, semantically equivalent descriptions may differ in surface form (Wang et al., 2022). Directly treating raw strings as discrete outcomes would therefore underestimate agreement across samples. We instead estimate empirical probabilities over semantic clusters: each free-form description is embedded using a Qwen-0.6B sentence encoder (Yang et al., 2025) and clustered via agglomerative clustering with a fixed cosine-distance threshold of 0.3. Each cluster is treated as a discrete latent outcome, yielding semantic identifiers $I_s^{(m)}$ and $C_s^{(m)}$ for estimating the empirical distributions of intentions and state changes. Appendix H further verifies that the aggregate uncertainty features are stable across different embedding encoders and clustering thresholds.

### 3.4 Prefix Risk Prediction

The micro and macro feature families provide complementary views of agent uncertainty. We now combine them into a unified prefix-level risk pre-

dictor, as illustrated in Figure 1.

We construct the full prefix feature vector by concatenation:

$$\mathbf{x}_t = [\mathbf{x}_t^{\text{micro}}; \mathbf{x}_t^{\text{macro}}] \in \mathbb{R}^{d_x}, \tag{5}$$

where $d_x$ denotes the total number of prefix-level features. A risk predictor maps this feature vector directly to a failure probability:

$$\hat{r}_t = \mathcal{F}_\theta(\mathbf{x}_t), \tag{6}$$

where $\hat{r}_t \in [0, 1]$ is interpreted as the estimated probability of eventual task failure conditioned on the current prefix.

Inspired by prior work on trajectory-level agent uncertainty calibration (Zhang et al., 2026b), we instantiate $\mathcal{F}_\theta$ as an L2-regularized logistic regression model, which provides a lightweight and directly probabilistic risk predictor while retaining feature-level interpretability through its learned coefficients. Given a prefix feature vector $\mathbf{x}_t \in \mathbb{R}^{d_x}$, the predicted failure risk is

$$\hat{r}_t = \mathcal{F}_\theta(\mathbf{x}_t) = \sigma(\mathbf{w}^\top \mathbf{x}_t + b), \tag{7}$$

where $\sigma(z) = 1/(1 + \exp(-z))$, $\mathbf{w} \in \mathbb{R}^{d_x}$ and $b \in \mathbb{R}$ are learned parameters. The model is trained by minimizing the regularized binary cross-entropy objective.

**Prefix-Level Failure Labeling.** A failed trajectory does not imply that every prefix step is erroneous. Assigning the final failure label to all prefixes would introduce a label mismatch, since early prefixes may still contain correct decisions and remain recoverable. We therefore identify a *key failure step* for each failed trajectory, defined as the earliest step after which the trajectory becomes unlikely to recover and eventually fails. Prefixes before this key step are labeled as non-failure ($y_{i,k} = 0$), while the key step and all subsequent prefixes are labeled as failure ($y_{i,k} = 1$); all prefixes from successful trajectories are labeled as non-failure. The key step is annotated using an LLM-as-judge protocol inspired by AgentRx (Barke et al., 2026b) following agent failure taxonomy (Zhu et al., 2025), where the judge inspects the full execution trajectory and selects the earliest critical failure step with supporting evidence. We further conduct manual auditing on a sampled subset of annotations to verify the plausibility of the identified key steps.

**Training details.** We construct training instances from prefix steps, where the input is the prefix-level feature vector $\mathbf{x}_{i,k}$ extracted from the execution history at step $k$ of trajectory $i$. Each instance is assigned a binary label $y_{i,k} \in \{0, 1\}$, indicating whether the specific prefix sequence constitutes a failure (1) or a success (0). We use 5-fold cross-validation split by task, ensuring that all prefixes from the same task appear in the same fold. To reduce dependence on trajectory length, count-based macro features are normalized by the observed prefix length, so the predictor captures behavioral rates such as repetition, looping, and execution-error frequency rather than raw accumulated counts. Features are standardized using statistics computed only from the training fold and then applied to the corresponding test fold.

# 4 Experimental Study

## 4.1 Experiment Settings

**Benchmarks and Datasets.** We evaluate our method on two web-agent benchmarks covering both controlled and real-world environments. The first is **WebArena-Lite** (Liu et al., 2024), which is deployed in filtered Docker-based environments and provides a controlled setting for reproducible web interaction. The second is **Online-Mind2Web** (Xue et al., 2025), which consists of real-world websites and therefore presents higher environment variability and grounding difficulty.

**Agent and Agent Backbones.** We use a ReAct-based (Yao et al., 2022) agent framework and study both open-source (**Qwen3-VL-30B** (Bai et al., 2025) and **Kimi 2.5** (Team et al., 2026)) and closed-source (**GPT-5.2**, **GPT-5.4-nano**, and **Claude Sonnet 4.6**) vision-language models as backbones for web agents.

**Baseline Methods.** We compare our method against several representative uncertainty and confidence baselines: (1) **Verbalized Confidence** (Tian et al., 2023): the agent is prompted to directly report its confidence in completing the task, evaluated both at step $k$ and as prefix-aggregated accuracy. (2) **Decision Entropy**: entropy of the next-step action distribution at step $k$, together with prefix-aggregated variants (mean over steps). (3) **HTC-Full** (Zhang et al., 2026b): a prefix-level adaptation of Holistic Trajectory Calibration that extracts calibration features from the observed prefix $\tau_{\leq k}$, including cross-step confidence dynamics, intra-step token-level stability, positional indicators, and structural trajectory attributes. (4) **Log-probability based methods**: token-level NLL at step $k$ and

| Model | Methods | Webarena-Lite | | | Mind2web | | |
|---|---|---|---|---|---|---|---|
| | | E-AURC↓ | BRIER↓ | AUROC↑ | E-AURC↓ | BRIER↓ | AUROC↑ |
| Claude | Verb. conf @k | 0.175 | **0.159** | 0.653 | 0.273 | 0.250 | 0.628 |
| | Verb. accu | 0.186 | 0.197 | 0.668 | 0.244 | 0.250 | **0.671** |
| | Act. Ent. @k | 0.371 | 0.251 | 0.585 | 0.367 | 0.232 | 0.534 |
| | Act. Ent. accu | 0.267 | 0.252 | 0.647 | 0.309 | **0.217** | 0.593 |
| | Macro (Ours) | 0.149 | 0.214 | 0.748 | **0.231** | 0.234 | 0.670 |
| | Micro (Ours) | 0.145 | 0.214 | 0.760 | 0.255 | 0.248 | 0.627 |
| | M&M (Ours) | **0.141** | 0.207 | **0.767** | 0.236 | 0.238 | 0.656 |
| GPT-5.2 | Verb. conf @k | 0.264 | 0.241 | 0.584 | 0.266 | 0.250 | 0.590 |
| | Verb. accu | 0.244 | 0.255 | 0.549 | 0.263 | 0.250 | 0.601 |
| | Act. Ent. @k | 0.319 | 0.242 | 0.566 | 0.334 | 0.216 | 0.519 |
| | Act. Ent. accu | 0.273 | 0.241 | 0.598 | 0.298 | 0.191 | 0.556 |
| | HTC Full | 0.179 | 0.238 | 0.670 | 0.228 | **0.154** | 0.653 |
| | Macro (Ours) | 0.159 | **0.200** | **0.784** | **0.203** | 0.227 | **0.683** |
| | Micro (Ours) | 0.195 | 0.215 | 0.730 | 0.222 | 0.233 | 0.655 |
| | M&M (Ours) | **0.154** | 0.200 | 0.775 | 0.206 | 0.226 | 0.678 |
| GPT-5.4 Nano | Verb. conf @k | 0.253 | 0.250 | 0.549 | 0.210 | 0.250 | 0.608 |
| | Verb. accu | **0.185** | 0.250 | 0.589 | 0.199 | 0.250 | 0.624 |
| | Act. Ent. @k | 0.265 | 0.218 | 0.500 | 0.206 | 0.127 | 0.569 |
| | Act. Ent. accu | 0.287 | 0.215 | 0.454 | 0.169 | **0.089** | 0.676 |
| | HTC Full | 0.315 | 0.270 | 0.571 | 0.238 | 0.120 | 0.574 |
| | Macro (Ours) | 0.210 | 0.185 | 0.728 | 0.127 | 0.185 | 0.798 |
| | Micro (Ours) | 0.236 | 0.195 | 0.660 | 0.151 | 0.199 | 0.758 |
| | M&M (Ours) | 0.192 | **0.175** | **0.748** | **0.121** | 0.181 | **0.805** |
| Qwen 30B | Verb. conf @k | 0.182 | 0.250 | 0.575 | 0.187 | 0.250 | 0.614 |
| | Verb. accu | 0.274 | 0.250 | 0.576 | 0.164 | 0.250 | 0.661 |
| | Act. Ent. @k | 0.187 | 0.223 | 0.436 | 0.175 | 0.118 | 0.554 |
| | Act. Ent. accu | 0.187 | 0.192 | 0.533 | **0.149** | **0.069** | 0.653 |
| | HTC Full | 0.187 | 0.189 | 0.737 | 0.173 | 0.086 | 0.723 |
| | Macro (Ours) | 0.110 | 0.146 | 0.856 | 0.156 | 0.182 | **0.783** |
| | Micro (Ours) | 0.106 | **0.127** | 0.860 | 0.176 | 0.192 | 0.740 |
| | M&M (Ours) | **0.102** | 0.134 | **0.869** | 0.159 | 0.184 | 0.777 |
| Kimi 2.5 | Verb. conf @k | 0.196 | 0.251 | 0.666 | 0.246 | 0.250 | 0.619 |
| | Verb. accu | 0.214 | 0.250 | 0.676 | 0.280 | 0.250 | 0.588 |
| | Act. Ent. @k | 0.276 | 0.244 | 0.617 | 0.295 | 0.198 | 0.564 |
| | Act. Ent. accu | 0.284 | 0.244 | 0.674 | 0.291 | 0.181 | 0.598 |
| | HTC Full | 0.175 | 0.208 | 0.748 | 0.261 | **0.164** | 0.632 |
| | Macro (Ours) | **0.112** | 0.181 | 0.831 | 0.206 | 0.219 | 0.706 |
| | Micro (Ours) | 0.146 | 0.197 | 0.778 | **0.200** | 0.218 | 0.710 |
| | M&M (Ours) | 0.112 | **0.177** | **0.833** | 0.202 | 0.216 | **0.714** |

Table 1: Performance comparison. (↓) lower is better, (↑) higher is better. M&M denotes Macro&Micro. **Bold** indicates the best; Underline indicates the second best.

prefix-aggregated variants including action logprob, output logprob mean, and accumulated output logprob (see Appendix E). (5) **Feature Ablation Baselines**: **micro-only** and **macro-only** variants of our method, used to evaluate the individual contribution of each feature family.

HTC-Full and other Log-probability based methods are unavailable for Claude as it doesn't support log-probability return. Temperature Scaling (Guo et al., 2017) is applied to all baseline methods (except HTC-Full) for fair comparison with our calibrated predictor.

**Evaluation Metrics.** We evaluate from three complementary perspectives. First, we measure calibration quality using **Brier Score (BS)** (Glenn et al., 1950).Second, we evaluate selective risk-control utility using **E-AURC** (Geifman et al., 2018), the excess area under the risk–coverage curve relative to the oracle ranking. Lower E-AURC indicates that the model more effectively ranks failure-prone prefixes as high risk, enabling low-risk prefixes to be retained while risky executions are abstained from or intervened upon. Third, we report **AUROC** as a standard discrimination metric measuring whether failed trajectories receive higher risk scores than successful ones.

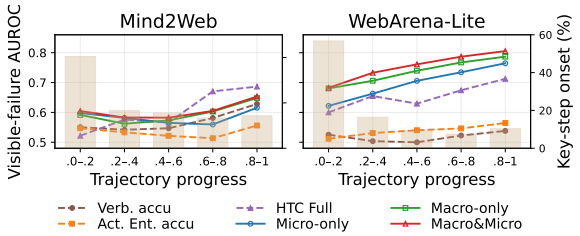


Figure 2: Key-step-aware prefix visibility curves. AUROC for detecting visible failure; bars indicate the first positive key-step position. Solid lines: ours; dashed: baselines.

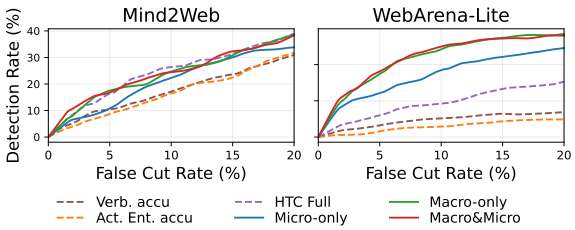


Figure 3: Detection–false-cut tradeoff under the key-step-aware early-cut policy.

### 4.2 Main Results

Table 1 shows that the proposed features mainly improve *risk discrimination and selective ranking*. On **WebArena-Lite**, Macro&Micro is the strongest overall variant: it is top-two in E-AURC for all five backbones and gives the best AUROC for four of them. The only AUROC exception is GPT-5.2, where Macro-only is slightly higher (0.784 vs. 0.775), suggesting that cumulative trajectory symptoms alone can already be highly predictive in this controlled environment. Micro-only is less consistent, but it is useful when local intention–action uncertainty is informative, as in the Qwen 30B WebArena-Lite setting where it gives the best Brier score and second-best AUROC.

The pattern on **Mind2Web** is more mixed but still supports the value of structured features. Our variants give the best E-AURC in four of five backbones and the best AUROC in four of five, with Macro or Macro&Micro usually outperforming confidence and entropy baselines. The exceptions are informative: Claude has a near tie between verbalized accuracy and Macro on AUROC (0.671 vs. 0.670), and Qwen 30B favors accumulated action entropy for E-AURC while Macro remains best on AUROC. This suggests that open-web failures are less uniformly expressed through a single signal; Macro features provide the most stable standalone evidence, while Micro features help when local decision uncertainty aligns with downstream failure.

Brier score shows a different trend. On WebArena-Lite, our methods often calibrate well, with Macro&Micro or Micro giving the best Brier score for four backbones. On Mind2Web, however, simpler confidence-, entropy-, or HTC-based baselines often achieve lower Brier scores. This separation is expected: E-AURC and AUROC evaluate whether risky prefixes are ordered correctly, while Brier score penalizes absolute probability error. Overall, the table indicates that the dual-level representation is strongest as a portable *risk-ranking* signal, while probability calibration remains more sensitive to benchmark and backbone.

### 4.3 Prefix-Time Analysis

We ask when failure becomes discriminable from a trajectory prefix, before considering any intervention policy. For a failed trajectory with first positive key-step label $k_i^\star$, prefixes before $k_i^\star$ are treated as negative and prefixes at or after $k_i^\star$ as positive; successful trajectories are always negative:

$$y_{i,k}^{\text{vis}} = \mathbb{1}[\text{trajectory } i \text{ fails and } k \geq k_i^\star]. \quad (8)$$

We divide each trajectory into five relative-time buckets and report AUROC for this visible-failure target. AUROC is used here because the question is separability; the next subsection uses false-cut-controlled detection to evaluate intervention utility.

Figure 2 shows that Macro&Micro detects visible failure earlier than the alternatives on average, with AUROC $0.651/0.684/0.729$ in the early/mid/late regions and the first averaged bin above $0.65$ at $0.2$–$0.4T$. The gain is driven mainly by **WebArena-Lite**, where Macro&Micro is strong from the first $40\%$ of execution (Early AUROC $0.708$) and reaches $0.805$ late. **Mind2Web** is noisier: Macro&Micro is best early ($0.594$), but HTC Full is stronger in the middle and late regions ($0.618/0.686$). This indicates that structured environments expose trajectory-level failure symptoms earlier, while open-web failures become reliably separable only later.

### 4.4 Selective Prediction and Early-Cut Simulation

We next test whether prefix risk can support an early stopping policy. A trajectory is cut once its risk exceeds a threshold before the final step. For failed trajectories, a cut counts as detected only if it occurs at or after the first positive key-step $k^\star$;

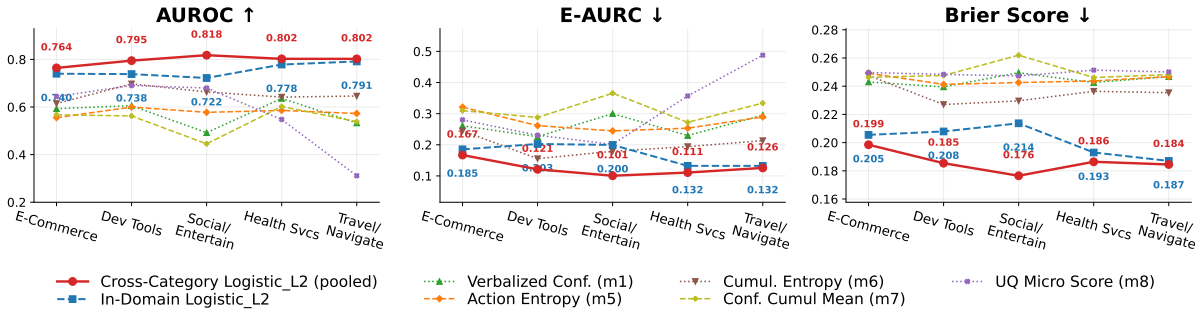


Figure 4: Task-regime generalization under key-step labels. We hold out one website regime for testing and train on the remaining regimes. The pooled Macro–Micro Logistic-L2 predictor remains strong across AUROC, E-AURC, and Brier score, indicating that the learned risk signals transfer across website categories.

successful trajectories contribute to False Cut Rate if they are cut at all:

$$\text{Det.} = \frac{\#\{\text{failed trajs. cut at } k \in [k^\star, T)\}}{\#\{\text{failed trajs.}\}}, \quad (9)$$

$$\text{FCR} = \frac{\#\{\text{successful trajs. cut at } k < T\}}{\#\{\text{successful trajs.}\}}. \quad (10)$$

We report Detection Rate under fixed false-cut budgets and compare against Verbalized Accuracy, Action Entropy Accuracy, and HTC Full.

Figure 3 show the same benchmark split. On **WebArena-Lite**, Macro&Micro is best at 5% and 10% FCR (32.7%, 47.6%), while Macro-only slightly leads at 20% FCR (57.1%). On **Mind2Web**, Macro-only is best only at 5% FCR (16.5%); HTC Full becomes strongest at 10% and 20% FCR (26.1%, 38.6%). Averaged across runs, Macro&Micro is best in the low-false-cut regime, while Macro-only becomes strongest when the intervention budget is relaxed.

### 4.5 Task-Regime Generalization

To evaluate the robustness and portability of our proposed dual-level features, we conduct a **cross-website generalization study** using key-step labels. We group tasks into five website regimes—E-Commerce, Developer Tools, Social/Entertainment, Health Services, and Travel/Navigation—and train a pooled Logistic-L2 risk predictor on all regimes except the held-out one. We compare it with an in-domain Logistic-L2 reference and single-signal baselines based on verbalized confidence, action entropy, cumulative confidence, cumulative entropy, and the UQ Micro score.

Figure 4 shows that the pooled Macro–Micro predictor transfers consistently across task regimes. It achieves the best AUROC in every held-out regime (0.764–0.818), while also giving the lowest E-AURC and Brier score across all five regimes. The gap is especially clear on Social/Entertainment, where the pooled model reaches AUROC 0.818, E-AURC 0.101, and Brier 0.176. In contrast, single-signal baselines are unstable across regimes: some confidence or entropy signals work on one category but degrade sharply on another. These results indicate that the dual-level feature representation captures risk patterns that are not specific to a single website type thus proving the strong generalization ability of our proposed framework.

## 5 Conclusion

We present a prefix-level risk predictor for web agents that estimates task-failure risk during execution. The predictor combines Macro features that capture trajectory-level behavioral patterns with Micro features that expose step-level intention–action–expectation mismatch. Across WebArena-Lite and Online Mind2Web with five backbones, these structured features improve risk ranking over confidence and entropy baselines, especially on E-AURC and AUROC. The results show that reliable web-agent risk prediction benefits from looking both across the trajectory prefix and inside individual decisions, while remaining interpretable enough to support analysis, selective prediction, and early-cut simulation.

## Limitations

Several limitations of the current work should be noted. First, the advantage of combining Macro and Micro features is not uniform across all settings: on certain benchmark–backbone combinations, the Macro-only variant performs competitively, and simpler baselines can remain competi-

tive on calibration-oriented metrics such as Brier score in some cases. Second, the current framework relies on repeated sampling ($N = 10$) and structured feature extraction, which introduces extra computational cost during online deployment. Third, the key-step annotation study, while validated with human-LLM agreement, relies primarily on LLM-based annotations and may not capture the full nuance of human expert judgment. These findings suggest that while structured risk prediction is a promising direction for web-agent uncertainty estimation, improving efficiency, robustness, and transferability across broader agent settings remains important future work.

## References


Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Yavuz Faruk Bakman, Duygu Nur Yaldiz, Baturalp Buyukates, Chenyang Tao, Dimitrios Dimitriadis, and Salman Avestimehr. 2024. Mars: Meaning-aware response scoring for uncertainty estimation in generative llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7752–7767.

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. 2026a. Agentrx: Diagnosing ai agent failures from execution trajectories. *Preprint*, arXiv:2602.02475.

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. 2026b. Agentrx: Diagnosing ai agent failures from execution trajectories. *arXiv preprint arXiv:2602.02475*.

Longchao Da, Xiaoou Liu, Jiaxin Dai, Lu Cheng, Yaqing Wang, and Hua Wei. 2025. Understanding the uncertainty of LLM explanations: A perspective based on reasoning topology. In *Second Conference on Language Modeling*.

Jinhao Duan, James Diffenderfer, Sandeep Madireddy, Tianlong Chen, Bhavya Kailkhura, and Kaidi Xu. 2025. Uprop: Investigating the uncertainty propagation of llms in multi-step agentic decision-making. *arXiv preprint arXiv:2506.17419*.

Yonatan Geifman, Guy Uziel, and Ran El-Yaniv. 2018. Bias-reduced uncertainty estimation for deep neural classifiers. *arXiv preprint arXiv:1805.08206*.

W Brier Glenn and 1 others. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Weijian Qi, Andrei Kopanev, Botao Yu, Bernal Jiménez Gutiérrez, Yiheng Shu, and 1 others. 2025. Mind2web 2: Evaluating agentic search with agent-as-a-judge. *arXiv preprint arXiv:2506.21506*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. 2024. Semantic entropy probes: Robust and cheap hallucination detection in llms. *arXiv preprint arXiv:2406.15927*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*.

Nicholas Lee, Lutfi Eren Erdogan, Chris Joseph John, Surya Krishnapillai, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2026. Agentic test-time scaling for webagents. *arXiv preprint arXiv:2602.12276*.

Yinghao Li, Rushi Qiang, Lama Moukheiber, and Chao Zhang. 2025. Language model uncertainty quantification with attention chain. In *Second Conference on Language Modeling*.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, and 1 others. 2024. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*.

Andrey Malinin and Mark Gales. 2020. Uncertainty estimation in autoregressive structured prediction. *arXiv preprint arXiv:2002.07650*.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 9004–9017.

Changdae Oh, Seongheon Park, To Eun Kim, Jiatong Li, Wendi Li, Samuel Yeh, Xuefeng Du, Hamed Hassani, Paul Bogdan, Dawn Song, and 1 others. 2026. Uncertainty quantification in llm agents: Foundations, emerging challenges, and opportunities. *arXiv preprint arXiv:2602.05073*.

Litu Ou, Kuan Li, Huifeng Yin, Liwen Zhang, Zhongwang Zhang, Xixi Wu, Rui Ye, Zile Qiao, Pengjun Xie, Jingren Zhou, and 1 others. 2025. Browseconf:

Confidence-guided test-time scaling for web agents. *arXiv preprint arXiv:2510.23458*.

Xin Qiu and Risto Miikkulainen. 2024. Semantic density: Uncertainty quantification for large language models through confidence measurement in semantic space. *Advances in neural information processing systems*, 37:134507–134533.

Shuchang Tao, Liuyi Yao, Hanxing Ding, Yuexiang Xie, Qi Cao, Fei Sun, Jinyang Gao, Huawei Shen, and Bolin Ding. 2024. When to trust llms: Aligning confidence with response quality. *arXiv preprint arXiv:2404.17287*.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, and 1 others. 2026. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. 2025. An illusion of progress? assessing the current state of web agents. *arXiv preprint arXiv:2504.01382*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Boxuan Zhang and Ruqi Zhang. 2025. Cot-uq: Improving response-wise uncertainty quantification in llms with chain-of-thought. *arXiv preprint arXiv:2502.17214*.

Jiaxin Zhang, Prafulla Kumar Choubey, Kung-Hsiang Huang, Caiming Xiong, and Chien-Sheng Wu. 2026a. Agentic uncertainty quantification. *Preprint*, arXiv:2601.15703.

Jiaxin Zhang, Caiming Xiong, and Chien-Sheng Wu. 2026b. Agentic confidence calibration. *arXiv preprint arXiv:2601.15778*.

Lingfeng Zhang, Jinpeng Hu, Hui Ma, Kuien Liu, Zenglin Shi, Meng Wang, and 1 others. 2026c. Webuncertainty: Dual-level uncertainty driven planning and reasoning for autonomous web agent. *arXiv preprint arXiv:2604.17821*.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, and 1 others. 2025. Where llm agents fail and how they can learn from failures. *arXiv preprint arXiv:2509.25370*.

## A Generative AI Usage Statement

Generative AI tools were used to assist with grammar checking and text polishing. No original ideas, experimental data, or core content were generated by AI. The authors take full responsibility for the accuracy and integrity of this work.

## B Error Annotation Methodology

We adopt a structured error annotation taxonomy adapted from AgentRx (Barke et al., 2026a), which categorizes agent failures along multiple dimensions including perception errors, reasoning errors, grounding errors, and execution errors.

**Annotation procedure.** We first collect agent trajectories from both WebArena-Lite and Online Mind2Web across all five backbone models. A subset of trajectories is manually annotated by the authors to establish ground-truth failure categories. Then all trajectories are annotated using an LLM-based annotation pipeline, where the LLM is provided with the full trajectory, the task specification, and the annotation taxonomy. The LLM annotator identifies whether the trajectory succeeded or failed, and for failed trajectories, assigns one or more fine-grained error categories to each step. In total, we annotate 3,364 trajectories. A detailed breakdown of failure mode frequencies is provided in Table 2

## C Empirical Observation Analysis

Table 3 (main text) identifies two structurally distinct failure patterns that each account for roughly one quarter to one third of all annotated failures. Below we define each constituent category and provide a concrete example drawn from the annotated trajectories.

### Observation 1: Error Behavioral Patterns

Failures in this group arise not from errors in the action itself, but from the agent's inability to correctly perceive, remember, or reason about its own

| Failure Cause | WA-Lite # | WA-Lite % | Mind2Web # | Mind2Web % |
|---|---|---|---|---|
| **Action** | **307** | **29.8** | **718** | **30.8** |
| Parameter Error | 255 | 24.8 | 631 | 27.0 |
| Planning–Action Disconnect | 11 | 1.1 | 44 | 1.9 |
| Format Error | 41 | 4.0 | 43 | 1.8 |
| **Planning** | **382** | **37.1** | **766** | **32.8** |
| Inefficient Planning | 255 | 24.8 | 408 | 17.5 |
| Constraint Ignorance | 103 | 10.0 | 253 | 10.8 |
| Impossible Action | 24 | 2.3 | 105 | 4.5 |
| **Reflection** | **272** | **26.4** | **645** | **27.6** |
| Progress Misassessment | 217 | 21.1 | 445 | 19.1 |
| Outcome Misinterpretation | 51 | 5.0 | 179 | 7.7 |
| Hallucination | 4 | 0.4 | 19 | 0.8 |
| Causal Misattribution | – | – | 2 | 0.1 |
| **Memory** | **40** | **3.9** | **75** | **3.2** |
| Retrieval Failure | 36 | 3.5 | 70 | 3.0 |
| Hallucination | 3 | 0.3 | 5 | 0.2 |
| Over-Simplification | 1 | 0.1 | – | – |
| **System** | **29** | **2.8** | **130** | **5.6** |
| Environment Error | 10 | 1.0 | 88 | 3.8 |
| Step Limit Exhaustion | 11 | 1.1 | 26 | 1.1 |
| Tool Execution Error | 8 | 0.8 | 16 | 0.7 |
| **Total** | **1,030** | 100.0 | **2,334** | 100.0 |

Table 2: Distribution of failure causes across WebArena-Lite and Online Mind2Web.

past behavior—leading to repetitive or prematurely abandoned trajectories.

**Reflection: Progress Misassessment.** The agent misjudges whether the task is advancing, which causes either premature termination (giving up when a retry would succeed) or futile continuation (repeating failed actions). *Example: after a failed click the page shows the same content; the agent concludes "task is stuck" and stops, though a different selector would have worked.*

**Memory: Retrieval Failure.** The agent fails to recall or re-use information that was already observed or stored earlier in the trajectory. Concretely, it often retrieves and re-executes a previously failed action. *Example: the agent found the target phone number 5 steps ago, but now searches the page from scratch instead of reusing the stored value.*

**Reflection: Causal Misattribution.** The agent correctly identifies that a failure occurred, but attributes it to the wrong cause, leading to an incorrect recovery strategy. *Example: a page fails to load due to a network timeout; the agent concludes the URL must be invalid and switches to a different website.*

**Observation 2: Intention–Action–Expectation Mismatch**

Failures in this group occur when the agent's intended action, the executed action, and the expected outcome are misaligned. The agent either executes the wrong operation, produces an unparseable command, or acts inconsistently with its own stated plan.

**Action: Parameter Error.** The agent selects the correct action type (e.g., `click`) but specifies the wrong target or arguments. *Example: the agent intends to click "Store Finder" but clicks a visually adjacent search icon instead.*

**Action: Planning–Action Disconnect.** The agent's verbally stated plan is correct, but the action it executes does not carry out that plan. *Example: the agent says "I should scroll to find the product," then clicks a sidebar link without scrolling.*

**Action: Format Error.** The agent produces malformed action syntax that the executor cannot parse. *Example: the agent outputs click("search") instead of the required click(selector="search")* .

# D Detailed Feature Definitions

## D.1 Micro Features

At each step $s$, we derive six scalar uncertainty indicators from the empirical latent decision distribution $\hat{p}_s(i, c, d \mid X_s)$:

1. **Intention Entropy** $H(I_s)$: entropy of the marginal intention distribution $\hat{p}_s(i)$.
2. **Decision Entropy** $H(D_s)$: entropy of the marginal decision distribution $\hat{p}_s(d)$.
3. **State-Change Entropy** $H(C_s)$: entropy of the marginal state-change distribution $\hat{p}_s(c)$.
4. **Conditional Decision Entropy** $H(D_s \mid I_s)$: expected entropy of the decision given the intention, capturing grounding-related uncertainty.
5. **Conditional State-Change Entropy** $H(C_s \mid D_s, I_s)$: expected entropy of the state change given intention and decision.
6. **Intention Contribution** $C_I = H(D_s) - H(D_s \mid I_s)$: mutual information between intention and decision, measuring how much knowing the sub-goal reduces action uncertainty.

| Structural Pattern | WA-Lite # | WA-Lite % | Mind2Web # | Mind2Web % | Total # | Total % |
|---|---|---|---|---|---|---|
| Intention–Action–Outcome Mismatch (Obs. 2) | 307 | 29.8 | 718 | 30.8 | 1,025 | 30.5 |
| Action: Parameter Error | 255 | 24.8 | 631 | 27.0 | 886 | 26.3 |
| Action: Planning–Action Disconnect | 11 | 1.1 | 44 | 1.9 | 55 | 1.6 |
| Action: Format Error | 41 | 4.0 | 43 | 1.8 | 84 | 2.5 |
| Error Behavioral Patterns (Obs. 1) | 253 | 24.6 | 517 | 22.2 | 770 | 22.9 |
| Reflection: Progress Misassessment | 217 | 21.1 | 445 | 19.1 | 662 | 19.7 |
| Memory: Retrieval Failure | 36 | 3.5 | 70 | 3.0 | 106 | 3.2 |
| Reflection: Causal Misattribution | – | – | 2 | 0.1 | 2 | 0.1 |
| **Combined (Obs. 1 + Obs. 2)** | **560** | **54.4** | **1,235** | **52.9** | **1,795** | **53.4** |
| **Total** | **1,030** | **100.0** | **2,334** | **100.0** | **3,364** | **100.0** |

Table 3: Two structurally distinct failure patterns emerge from the taxonomy. Together they account for approximately half of all failures in each environment.

Each indicator is aggregated over the prefix $\{0, \ldots, t\}$ using three operators:

- **Mean**: $\frac{1}{t+1} \sum_{s=0}^{t} f_s$
- **Maximum**: $\max_{s \leq t} f_s$
- **At-step**: $f_t$ (current value)

This yields $6 \times 3 = 18$ micro features.

### D.2 Macro Features

Macro features are prefix-level behavioral descriptors computed from the observed action sequence, execution metadata, and the agent's own reasoning text. Each row corresponds to a prefix ending at step $k$, so every feature only uses information from steps $0, \ldots, k$. The final feature matrix contains 31 macro columns:

**Prefix position.**

`step_k` The current zero-based step index.

**Action repetition and loop structure.**

`n_single_loops` Number of repeated single-action loop segments observed in the prefix.

`n_double_loops` Number of alternating two-action loop segments observed in the prefix.

`loop_length_current` Loop type at the tail of the trajectory: 0 for no loop, 1 for repeated same-action suffix, and 2 for alternating two-action suffix.

`n_distinct_actions` Number of distinct canonicalized actions observed up to step $k$.

`action_count_current` Number of times the current canonical action has appeared in the prefix.

`max_action_count` Maximum repetition count of any canonical action in the prefix.

`action_repeat_rate` Prefix length divided by the number of distinct canonical actions, measuring repetition density.

`steps_since_new_action` Number of steps since the most recent previously unseen canonical action appeared.

**Execution and grounding errors.**

`n_grounding_errors` Cumulative number of grounding-error steps. Mind2Web uses thought-derived grounding proxy signals; WebArena-Lite uses explicit grounding failures together with the same proxy signals.

`n_parse_errors` Cumulative number of parse-error actions.

`consecutive_errors_now` Length of the trailing error streak ending at step $k$, where each step is either a grounding error or a parse error.

`grounding_error_at_k` Indicator that the current step is a grounding error under the benchmark-specific definition.

`parse_error_at_k` Indicator that the current step is a parse-error action.

**grounding_loop_count** Number of contiguous grounding-error streaks of length at least two in the prefix.

**Action composition and target diversity.**

**n_click** Cumulative number of click actions.

**n_type** Cumulative number of type actions.

**n_scroll** Cumulative number of scroll actions.

**n_go_back** Cumulative number of go-back actions.

**n_distinct_elements** Number of distinct element identifiers targeted in the prefix.

**action_type_entropy** Entropy of the action-type distribution in the prefix.

**type_text_length** Cumulative number of typed characters.

**Thought-derived diagnostic signals.**

**thought_action_ineffective_at_k** Indicator that any reasoning sample at step $k$ explicitly says the chosen action or strategy was ineffective or had no effect.

**thought_action_ineffective_cumul_count** Cumulative count of steps where the reasoning text reports an ineffective action or strategy.

**thought_wrong_context_at_k** Indicator that any reasoning sample at step $k$ says the agent is on the wrong page, website, or workflow.

**thought_wrong_context_cumul_count** Cumulative count of steps where the reasoning text reports wrong context.

**thought_environment_error_at_k** Indicator that any reasoning sample at step $k$ describes an environment, website, application, or blocking-UI failure.

**thought_environment_error_cumul_count** Cumulative count of steps where the reasoning text describes an environment or UI failure.

**Reasoning length.**

**reasoning_len_at_k** Mean reasoning-text length at step $k$.

**reasoning_len_cumul_mean** Cumulative mean reasoning-text length over steps $0, \ldots, k$ with available reasoning text.

**reasoning_len_cumul_max** Cumulative maximum reasoning-text length over steps $0, \ldots, k$.

Unlike the Micro features, these Macro features are already defined at the prefix level and are not produced by repeated model sampling. Count-based Macro features are normalized by the observed prefix length before model fitting, as described in Section 3.3.

## E Confidence Baseline Details

We provide additional details for the log-probability baselines used in our experiments:

- **Action Logprob**: the mean log-probability of the tokens comprising the selected action.
- **Output Logprob Mean**: the mean token-level log-probability over the entire model output at step $k$.
- **Output Logprob Accumulated**: the accumulated product of token probabilities over all steps in the prefix.

All baselines are calibrated using Platt scaling on the same validation split as our method.

## F Efficiency Analysis

[If moved from main text: detailed efficiency analysis with sampling budgets $N \in \{3, 5, 7, 10\}$ across all backbones.]

## G Per-Model Threshold Analysis

[If moved from main text: detailed per-model threshold calibration results.]

## H Clustering Robustness

Because our Micro features are derived from semantic clustering, they could in principle be sensitive to the choice of encoder and clustering threshold. We test this directly on **WebArena-Lite / GPT-5.2** by varying the encoder across **Qwen**,

| Model | Methods | Webarena-Lite | | | Mind2web | | |
|---|---|---|---|---|---|---|---|
| | | E-AURC↓ | BRIER↓ | AUROC↑ | E-AURC↓ | BRIER↓ | AUROC↑ |
| GPT-5.2 | Action log | 0.338 | 0.253 | 0.614 | 0.264 | 0.205 | 0.600 |
| | O log mean | 0.206 | 0.250 | 0.550 | 0.293 | 0.168 | 0.562 |
| | O log accm | 0.234 | 0.250 | 0.550 | 0.267 | 0.160 | 0.594 |
| | Macro (Ours) | 0.159 | **0.200** | **0.784** | **0.203** | 0.227 | **0.683** |
| | Micro (Ours) | 0.195 | 0.215 | 0.730 | 0.222 | 0.233 | 0.655 |
| | M&M (Ours) | **0.154** | 0.200 | 0.775 | 0.206 | 0.226 | 0.678 |
| GPT-5.4 Nano | Action log | 0.309 | 0.249 | 0.527 | 0.227 | 0.144 | 0.582 |
| | O log mean | 0.229 | 0.255 | 0.535 | 0.262 | 0.131 | 0.496 |
| | O log accm | 0.237 | 0.262 | 0.545 | 0.258 | **0.119** | 0.506 |
| | Macro (Ours) | 0.210 | 0.185 | 0.728 | 0.127 | 0.185 | 0.798 |
| | Micro (Ours) | 0.236 | 0.195 | 0.660 | 0.151 | 0.199 | 0.758 |
| | M&M (Ours) | **0.192** | **0.175** | **0.748** | **0.121** | 0.181 | **0.805** |
| Qwen 30B | Action log | 0.225 | 0.247 | 0.408 | 0.236 | 0.206 | 0.459 |
| | O log mean | 0.236 | 0.250 | 0.297 | 0.275 | 0.175 | 0.297 |
| | O log accm | 0.246 | 0.247 | 0.305 | 0.275 | 0.119 | 0.301 |
| | Macro (Ours) | 0.110 | 0.146 | 0.856 | **0.156** | 0.182 | **0.783** |
| | Micro (Ours) | 0.106 | **0.127** | 0.860 | 0.176 | 0.192 | 0.740 |
| | M&M (Ours) | **0.102** | 0.134 | **0.869** | 0.159 | 0.184 | 0.777 |
| Kimi 2.5 | Action log | 0.276 | 0.250 | 0.509 | 0.350 | 0.241 | 0.494 |
| | O log mean | 0.258 | 0.250 | 0.602 | 0.282 | 0.161 | 0.585 |
| | O log accm | 0.256 | 0.250 | 0.647 | 0.249 | **0.157** | 0.615 |
| | Macro (Ours) | **0.112** | 0.181 | 0.831 | 0.206 | 0.219 | 0.706 |
| | Micro (Ours) | 0.146 | 0.197 | 0.778 | **0.200** | 0.218 | 0.710 |
| | M&M (Ours) | 0.112 | **0.177** | **0.833** | 0.202 | 0.216 | **0.714** |

Table 4: Performance comparison with log-probability based baselines. (↓) lower is better, (↑) higher is better. M&M denotes Macro&Micro. **Bold** indicates the best; Underline indicates the second best.

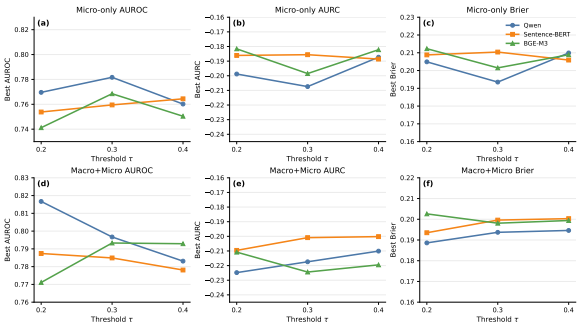


Figure 5: Clustering sensitivity of downstream prediction on WebArena-Lite / GPT-5.2. The six panels report the best **Micro-only** and **Macro+Micro** performance under different encoders and clustering thresholds $\tau$, evaluated with AUROC, AURC, and Brier score.

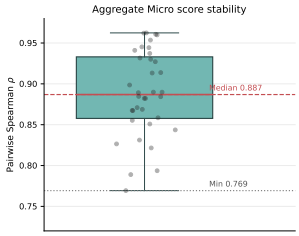


Figure 6: Aggregate Micro stability across clustering settings. Each point corresponds to one pair of clustering settings, evaluated by the Spearman correlation of the aggregate Micro score over aligned prefixes. Although local cluster assignments vary, the aggregate Micro ranking remains highly consistent.

**Sentence-BERT (MiniLM)**, and **BGE-M3**, and the agglomerative clustering threshold across $\tau \in \{0.2, 0.3, 0.4\}$, yielding nine clustering settings in total. Since Macro features do not depend on clustering, we show **Macro-only** only as a reference in the Micro-only AUROC panel, and focus the robustness analysis itself on **Micro-only** and **Macro+Micro**.

The local clustering statistics do shift across settings. In particular, quantities tied directly to cluster formation, such as $K_S$, $K_C$, and $U_{\text{sem}}$, vary noticeably. However, the *aggregate* uncertainty signal is much more stable. After aligning prefixes by (`task_id`, `step_k`), we build an aggregate Micro score by standardizing the 11 uncertainty indicators and averaging them within each setting. Across all 36 setting pairs, these aggregate scores remain highly consistent, with median Spearman correlation 0.887, median Pearson correlation 0.932, and minimum Spearman correlation 0.769 (Figure 6).

This robustness carries over to downstream risk prediction. Across the nine settings, the best **Micro-only** predictor attains AUROC between $0.741$ and $0.782$, AURC between $-0.238$ and $-0.169$, and Brier score between $0.170$ and $0.222$. For the **Macro+Micro** predictor, AUROC ranges from $0.771$ to $0.817$, AURC from $-0.240$ to $-0.166$, and Brier score from $0.164$ to $0.213$. Taken together, these results support the claim that *clustering choices change individual cluster labels more than they change the overall uncertainty ranking signal*, which remains stable enough to yield consistent downstream performance.