Network-Level Prompt and Trait Leakage in Local Research Agents

Hyejun Jeong Mohammadreza Teymoorianfard Abhinav Kumar
Amir Houmansadr Eugene Bagdasarian

University of Massachusetts Amherst

{hjeong,mteymoorianf,abhinavk,amir,eugene}@cs.umass.edu

Abstract

We show that Web and Research Agents (WRAs)—language-model-based systems that investigate complex topics on the Internet—are vulnerable to inference attacks by passive network observers. Deployment of WRAs *locally* by organizations and individuals for privacy, legal, or financial purposes exposes them to DNS resolvers, malicious ISPs, VPNs, web proxies, and corporate or government firewalls. However, unlike sporadic and scarce web browsing by humans, WRAs visit 70−140 domains per each request with a distinct timing pattern creating unique privacy risks.

Specifically, we demonstrate a novel prompt and user trait leakage attack against WRAs that only leverages their network-level metadata (i.e., visited IP addresses and their timings). We start by building a new dataset of WRA traces based on real user search queries and queries generated by synthetic personas. We define a behavioral metric (called OBELS) to comprehensively assess similarity between original and inferred prompts, showing that our attack recovers over 73% of the functional and domain knowledge of user prompts. Extending to a multi-session setting, we recover up to 19 of 32 latent traits with high accuracy. Our attack remains effective under partial observability and noisy conditions. Finally, we discuss mitigation strategies that constrain domain diversity or obfuscate traces, showing negligible utility impact while reducing attack effectiveness by an average of 29%.

1 Introduction

Web and Research Agents (WRAs) are reshaping how users interact with information. Unlike traditional assistants that answer isolated queries, these Large Language Model (LLM)-powered systems autonomously plan, browse, and synthesize knowledge across the web [31]. Major AI providers are integrating such agents directly into their flagship assistants or browsers, including OpenAI [58], Google Gemini [24], Grok [90], Mistral [44], and Perplexity [61]. Open-source counterparts such as LangChain's Local Deep Researcher [38], GPT Researcher [22], and Agent Laboratory [69] are also widely adopted in academic and developer communities. Their growing deployment in assistants, productivity platforms, and enterprise environments, particularly within privacy-sensitive sectors such as healthcare, law, and finance, raises urgent concerns about their security and privacy implications. While on-device execution through locally-deployable models and TLS-encrypted access could suggest privacy guarantees, this perception is misleading. Yet WRAs leave unavoidable, structured metadata traces that can reveal sensitive user intent, creating an underexplored leakage vector.

As autonomous agent systems, WRAs inherit vulnerabilities from both LLMs and traditional web browsing [8, 12, 32, 73]. Yet they also expose a distinct, underexplored vulnerability: behavioral traces determined by an LLM acting on a user's query. Prior work has shown that user intent and goals can be inferred from search queries and browsing activity [34, 66]. Even when content is encrypted, metadata such as domain names, access order, payload size, and timing can silently reveal what a user is trying to accomplish [55, 77].

Crucially, these behavioral traces are operationally distinguishable. WRAs generate dense cascades of semantically related domain visits within short time windows [61]. Unlike cognitively paced, irregular, and opportunistic human browsing [36], WRAs follow deterministic retrieval-and-synthesis loops with low variance in per-step timing. As a result, a single WRA session often traverses 40–100 distinct domains, whereas human sessions typically rely on few sites [85]. These structured, high-throughput access patterns make WRA traffic reliably separable from human activity and particularly amenable to profiling and inference attacks [16, 45].

We adopt a realistic threat model in which a passive adversary, (e.g., DNS resolver, ISP, enterprise firewall, or local network operator) observes only domain-level traffic metadata, without access to prompts, page contents, or model internals. Despite this limited visibility, we show that adversaries can reliably recover what users asked and who they are.

Code, prompts available at https://github.com/umass-aisec/wra

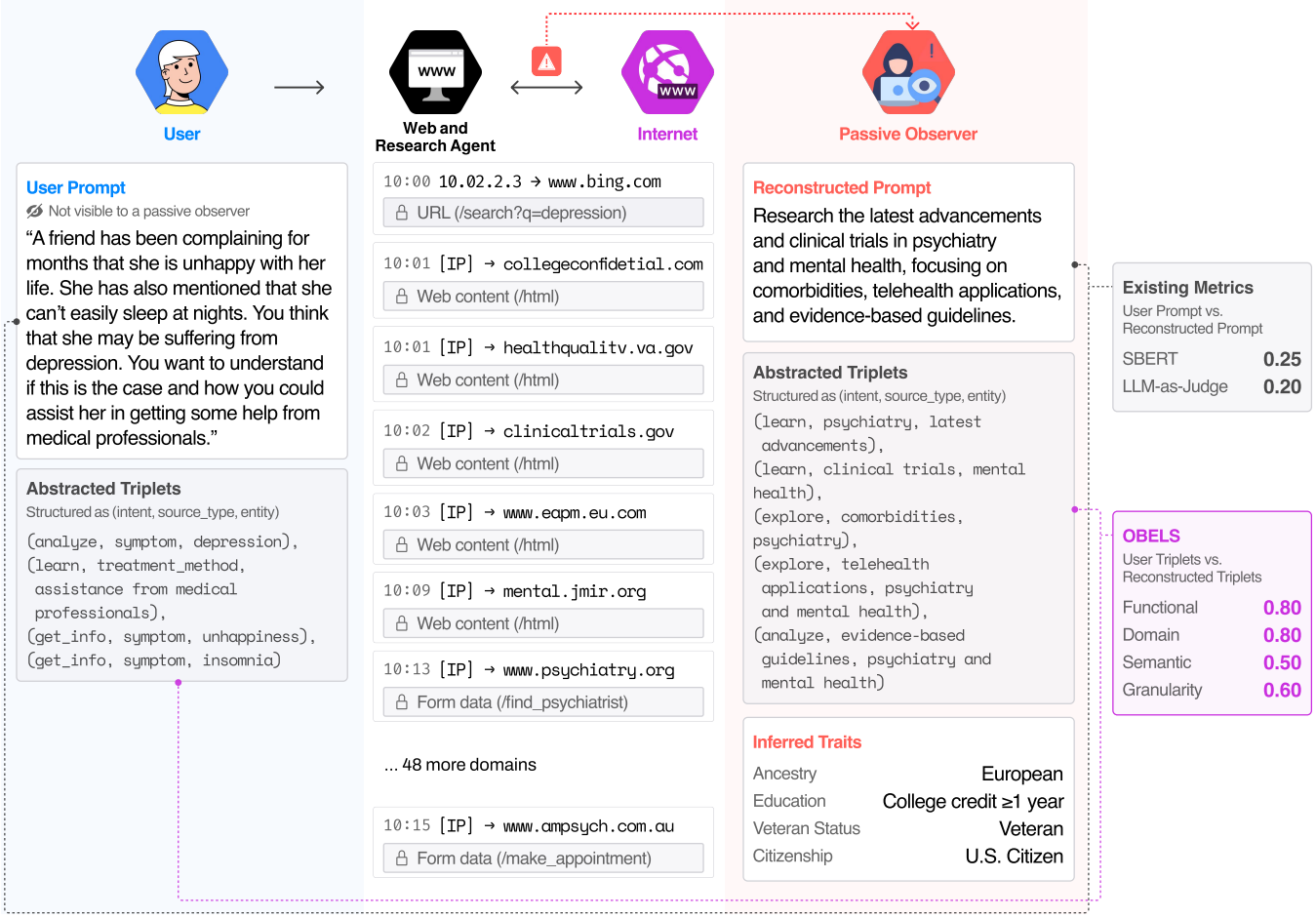

Figure 1: **Overview of the attack.**

Motivating Examples. Consider a user seeking reproductive health information: their research agent may visit clinics, support forums, and medical sites. Even without access to page contents, these traces alone can expose the agent's internal reasoning process and deeply stigmatized intent. As WRAs become integrated into daily decision-making, their behaviors present a new leakage channel. In essence, privacy failures emerge not from what agents say, but from how they act.

Our Work. We present privacy leakage attacks that exploit metadata traces of WRAs deployed locally. The attacks include **(1) Prompt Recovery**, which recovers a user's original prompt from observed domain sequences, and **(2) Trait Inference**, which profiles latent attributes (e.g., gender, ideology) from the agent's multi-session browsing patterns. The adversary trains an inference model using task-specific strategies such as In-Context Learning (ICL) and fine-tuning to map browsing traces to inference targets (Table 2). Our results show that the attacks remain effective even when 40% of the traces are masked or fully obfuscated, revealing that encrypted content alone cannot guarantee privacy (Table 10). We observe that WRAs visit dozens of marginal or redundant domains that do not improve the final report but still expand the adversary's visibility (Table 11). To mitigate these risks, we propose and evaluate practical defenses that either **hide traces**, by injecting LLM-generated decoy prompts and virtual personas to confuse inference (Section 8.1), or **block traces**, by redirecting tasks to multipurpose sources or LLM knowledge, avoiding uniquely identifying domains (Section 8.2).

Our contributions are as follows:

- **New attack surface:** We introduce metadata-based privacy attacks from behavioral traces produced by WRAs and propose a general two-stage pipeline for prompt recovery and trait inference.
- **Benchmark dataset and OBELS metric:** We construct and release a dataset linking domain traces to prompts and persona traits, and introduce OBELS, an ontology-aware metric for evaluating privacy leakage through behavioral traces.
- **Ablation and mitigation:** We conduct extensive experiments across agents, LLMs, and masking conditions, and evaluate mitigation that constrain domain diversity or obfuscate traces while negligibly affecting utility.

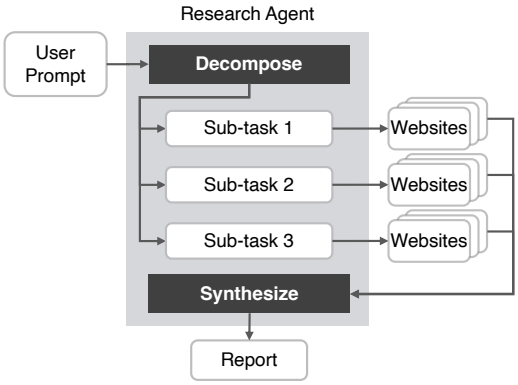

Figure 2: The workflow of Web and Research Agents.

2 Background and Related Work

2.1 AI, Web, and Research Agents

Standalone LLMs to AI Agents. LLMs excel at language understanding but are limited as static knowledge systems [2,80]. They cannot access real-time information, call external APIs, or execute multi-step plans, but produce responses based only on their pretrained knowledge. Early solutions such as RAG [25,41] and tool-augmented LLMs [15,63,67,72] improved grounding and tool use, but remain constrained to pre-indexed corpora and invoking tools only when explicitly instructed. AI agents address this by embedding LLMs into a perception–action loop [74,83,91], where the model plans, executes, and adapts across multiple steps, enabling autonomous reasoning and dynamic interaction with external environments.

Web Agents. Generic agents can reason about which queries to run, but often lack robust capabilities for controlling browsers, parsing dynamic content, handling JavaScript-heavy pages, or managing session state across multiple sites. Web agents integrate LLM planning with browser control: issuing search queries, navigating between sources, extracting content, and deciding which actions to perform based on the evolving context [16]. Frameworks like AutoGen [89] and Browser-Use [46] exemplify these capabilities, though evaluations often emphasize narrow navigation tasks.

Research Web Agents (WRAs). WRAs operationalize deep evidence synthesis through a *search–retrieve–extract–synthesize* loop across many sources (Figure 2). Given a user query, the agent generates sub-queries and issues them via APIs such as Tavily [79] to minimize token overhead and retrieve relevant documents efficiently. Retrieved content is parsed into evidence, distilled into notes, and evaluated to decide whether further exploration is needed. This process repeats until sufficient coverage is achieved, at which point the agent composes a comprehensive, citation-rich output [16, 31, 61, 68, 95]. Systems exemplifying this paradigm include GPT Researcher [22], LangChain's Local Deep Researcher [38], OpenAI Deep Research [58], Grok DeepSearch [90], and Gemini Deep Research [24].

Because WRAs generate dense browsing traces over dozens of domains, they amplify metadata-leakage risks. This risk has been underscored when Grok and OpenAI misconfigurations exposed private prompts to public indexing [43,76]. Locally deployed agents can be one of the mitigation strategies against such threats. Running on personal machines or servers, they preserve confidentiality while offering lower latency, offline use, and lower cost. Examples include GPT-Researcher [22] with self-hosted backends such as Ollama [56] and AMD's GAIA [3] to operate fully on-device, giving users more control over both their data and their agent workflows.

2.2 Privacy Attacks

Threats in LLMs. LLMs introduce unique privacy risks due to their training data and the ways users interact with them. One major class of attacks focuses on hidden-input inference, recovering sensitive inputs such as training data, system prompts, or user queries [11, 12, 32, 42, 60, 87, 94]. Other work has focused on recovering hidden system prompts used internally by providers [32,94], where adversaries probe model input and output patterns to elicit protected instructions. Unlike these approaches, our setting assumes *no direct access to the model* and relies instead on indirect behavioral signals.

Beyond direct extraction, studies show that semantic leakage occurs even when the attacker only observes generated outputs. Pasquini et al. [60] and Liu et al. [42] found that reconstructed prompts often preserve enough semantic content for intent inference, while Carlini et al. [11] and Weiss et al. [87] demonstrated that response timing and token-length patterns can reveal coarse-grained information about prompt topics. These attacks, however, assume access to model outputs or fine-grained response timings. By contrast, we show that **passively observable network-level metadata alone**, without accessing queries or responses, can be sufficient to infer hidden prompts.

LLMs also introduce privacy risks through trait inference and user profiling. Their outputs carry author-identifying signals [6, 29, 48, 71], allowing inference of attributes such as age, gender, and personality. Other risks include speculative decoding revealing fragments of private inputs [86] and long-context memory retaining user data across sessions for later retrieval [82]. Unlike these works, we assume a **strictly weaker threat model**: the adversary does **not** observe model inputs, outputs, or internal states, relying solely on external traffic traces (e.g., domain sequences, timing, payload sizes).

Threats in Web Privacy. Metadata leakage has long been recognized as a central privacy risk on the web. Classical work on anonymous communications (e.g., MIX Nets [14] and Tor [19]) demonstrated that encrypting content is insufficient

when traffic patterns remain exposed. Website Fingerprinting (WF) studies [9, 26, 35, 88] showed that adversaries can infer visited pages based on encrypted flow patterns, while large-scale deanonymization attacks like the Netflix attack [49] revealed that sparse behavioral traces are often enough to re-identify individuals. Subsequent work confirmed that traffic features, such as packet sizes, burst patterns, and inter-arrival timing, can fingerprint websites even under protections like Tor [27, 59]. Parallel work in web tracking exposed persistent profiling vectors, including cookie-based tracking [40], cookieless fingerprinting [1, 50], and evercookies [4].

Recent work has shifted to domain-level metadata as a source of behavioral leakage. Oliveira et al. [55] showed that as few as four domains can uniquely re-identify most users in real-world datasets. Crichton et al. [17] found that domain sequence fingerprints remain stable across years, even after cookie resets. Cross-context linkage is also possible: Naini et al. [47] and Su et al. [77] demonstrated that anonymized browsing logs can be matched across platforms, breaking assumed protections. Whereas classical WF identifies *which* site a user visits, our attack infers *user's task or intent* from agent-driven traversals. The two are complementary: WF reveals destinations, while our method reconstructs prompts or traits from the resulting domain sequences.

While protocol-level defenses such as HTTPS, DNS-over-HTTPS (DoH), and DNS-over-TLS (DoT) aim to hide domain names too, they often fail to conceal key identifiers. Hoang et al. [28] showed that TLS handshakes frequently leak domain information, while Sunahara et al. [78] pointed out widespread misconfigurations in encrypted DNS deployments, enabling persistent metadata leakage. Similarly, Wang et al. [84] demonstrated that in-app telemetry (e.g., navigation events, click logs, and tracking APIs) leaks fine-grained behavioral patterns to platforms. Unlike these works, we assume a weaker adversary: a passive network observer without privileged telemetry access, relying only on network-level domain sequences and packet metadata.

Agentic Privacy and Security Threats. AI agents amplify privacy risks by combining LLM reasoning with active browsing and external tool integration [20]. Kim et al. [37] showed that compromised agents can scrape PII from visited pages and reuse it in phishing, while InjecAgent [93] demonstrated that crafted payloads in webpages can trigger secret retrieval or unauthorized service access. Defenses such as AirGapAgent [7] and CHeaT [5] aim to counter these threats through context isolation, behavior cloaking, and honey-token traps.

However, these prior studies assume an adversary with visibility into the agent's queries, outputs, or injected prompts. In contrast, our work identifies a **new and underexplored attack surface**: even without access to agent inputs, outputs, or model internals, **domain sequences and packet-level metadata alone** can be exploited to reconstruct hidden prompts and infer sensitive user traits. This highlights a distinct dimension of privacy leakage introduced by web and research agents, where autonomous decision-making produces metadata-rich browsing patterns vulnerable to adversarial inference. To our knowledge, this is the first study to identify and exploit metadata-driven leakage in WRAs.

3 Overview: Prompt and Trait Leakage Attack

We study a *passive network adversary* that monitors encrypted traffic between the AI agent (acting on behalf of the user) and the internet. Without access to page contents or search queries, the adversary can exploit metadata leakage to recover sensitive information about the user's inputs and attributes.

Problem Definition. WRAs execute user prompts by issuing queries, following links, and retrieving information, leaving behind browsing traces that encode sensitive signals. Even if the underlying page contents or prompts are hidden, the sequence and diversity of domains can reveal the intent behind a query, while recurring patterns across sessions expose persistent user traits. An adversary has clear incentives to exploit these inferences: prompt recovery compromises user intent privacy, while trait inference enables long-term profiling that can be monetized, surveilled, or used for manipulation.

Prompt Recovery. With proxy dataset $D_{\text{PR}} = \{(WRA(p_j), p_j)\}$ consisting of pairs indexed by j, the adversary constructs an inference model M_{PR}. For a new prompt p with trace $t = WRA(p)$ (optionally filtered to $\tilde{t}$), the adversary recovers the prompt $\hat{p}$, aiming to maximize functional similarity:

$$\hat{p} = M_{\text{PR}}(\tilde{t}), \quad \max \ \text{Sim}\big(M_{\text{PR}}(\tilde{t}), p\big).$$

Trait Inference. Let τ denote a user's latent trait vector and $\{p_i\}_{i=1}^{N}$ the prompts across N sessions, yielding $t^{1:N} = WRA(\{p_i\}_{i=1}^{N})$. From proxy dataset $D_{\text{TI}} = \{(WRA(\{p_i^{(j)}\}_{i=1}^{N}), \tau_j)\}$, the adversary instantiates an inference model M_{TI}. On a new multi-session trace $t^{1:N}$ and infers trait $\hat{\tau}$, maximizing agreement:

$$\hat{\tau} = M_{\text{TI}}(\tilde{t}^{1:N}), \quad \max \ \text{Score}\big(M_{\text{TI}}(\tilde{t}^{1:N}), \tau\big).$$

where Sim and Score denote task-appropriate semantic similarity and trait-agreement metrics, respectively.

3.1 Threat Model

Adversary Goals. The adversary's objective is to infer sensitive user information from the browsing trace. We focus on two representative leakage goals that capture both short- and long-term risks. The first is **prompt recovery**, where the adversary seeks to infer queries functionally equivalent to the user's original input (e.g., detecting that a user asked about "signs of depression and seeking professional mental health support"). The second is **trait inference**, where the adversary profiles latent attributes such as gender, religion, and political

ideology by correlating patterns across multiple browsing sessions. Together, these illustrate how metadata can expose both immediate user intent and persistent personal characteristics.

Adversary Capabilities. The adversary observes domain-level metadata generated by the agent's web activity, including domain names (via plaintext DNS or TLS handshake metadata such as SNI), the order of domain visits, and coarse packet-level features such as timing and payload sizes. The adversary does *not* observe full URLs (e.g., query strings), page contents, user prompts, or the agent's intermediate queries.

In practice, visibility may be further reduced by Content Delivery Networks (CDNs), reverse proxies, or load balancers that aggregate multiple services under a small number of endpoints. To capture such weaker settings, we explicitly evaluate partial-visibility scenarios in which only a fraction of domains (e.g., 80%–10%) is observable (Table 11). These capabilities correspond to realistic on-path observers such as local ISPs, corporate firewalls, DNS resolvers, VPN providers, or shared network operators that collect encrypted metadata without decryption or manipulation.

Adversary Assumptions. We assume the user interacts with a local web or research agent that autonomously performs web browsing in response to a natural language prompt. The adversary is aware of the agent's general browsing capabilities, such as issuing web searches, clicking links, scrolling pages, or scraping structured data from web pages. The adversary monitors the session between the agent and the Internet but does not alter traffic. We treat full-session metadata visibility as an **upper bound**: weaker adversaries such as DNS resolvers or enterprise firewalls may see only subsets of domains, yet our experiments show that WRA traffic remains highly distinguishable due to its structured, high-volume access patterns. We further assume that DNS and TLS handshake metadata remain visible (i.e., no DoH, DoT, or ECH) and that users do not employ VPNs, Tor, or other anonymizing proxies that would conceal domain sequences completely.

3.2 Attack Scenarios

In this section, we illustrate when a passive adversary would actually deploy our prompt and trait leakage attack. Locally deployed WRAs are especially relevant in settings where users avoid reliance on remote APIs, reduce recurring costs, maintain control over proprietary workflows, or process highly confidential information that cannot be shared externally. In such cases, browsing traffic is generated directly on the client machine, exposing metadata to on-path observers.

Personal WRAs. An individual running a local WRA for personal tasks, such as health research, financial planning, or private study, may assume that confidentiality is preserved by avoiding cloud services. Yet even with local deployment, ISPs, enterprise firewalls, or other intermediaries can still observe traces and reconstruct prompts or infer private traits, enabling targeted advertising, discrimination, or phishing.

Academic WRAs. In university, students and faculty use local WRAs for daily tasks, coursework, or research, leaking information about politics or unpublished work. External observers like ISPs or competing research entities could exploit traces to reconstruct prompts or profile academic interests.

Industrial WRAs. At the company level, employees use local WRAs for competitor analysis, strategic planning (e.g., product roadmaps, M&A interests, financial planning), or handling confidential information (e.g., legal case files or patient records). These traces can be mined for business intelligence by competitors, ISPs, cloud providers, or even internal IT monitors. In startup companies, small teams could rely on local WRAs for cost savings or planning, but this may lead to leaks about funding strategies, partnership negotiations, or technical stack decisions. ISPs or data brokers could harvest and resell the information or insights to competitors.

State WRAs. At the government or state level, state-operated firewalls or regulators monitoring domestic traffic could reconstruct politically sensitive prompts or identify activists and vulnerable communities for closer surveillance.

Overall, local deployment makes browsing activity directly observable in ways that remote, server-side agents do not. For adversaries, such visibility creates actionable opportunities to reconstruct immediate user prompts or build long-term trait profiles, depending on the target and setting.

4 Attack Methodology

We present the end-to-end pipeline for our privacy attacks against AI agents. Both tasks—prompt recovery and trait inference—follow a shared two-stage process: (1) **offline model construction** on proxy datasets D_{PR} or D_{TI}, yielding M_{PR} or M_{TI}; and (2) **online attack execution** on real user traces t or $t^{1:N}$ to produce $\hat{p}$ or $\hat{\tau}$. The pipeline remains structurally consistent across tasks, though the input sources and inference targets differ.

Figure 3 illustrates our two-stage attack pipeline, separating the adversary's model construction from the attack execution. A two-stage design is necessary because the adversary cannot directly obtain a large set of real user prompts paired with their browsing traces. Instead, the adversary first trains an inference model on proxy data that mimics real-world interactions, and then applies this trained model to actual users.

4.1 Proxy Prompt Generation

To train the inference model in Stage 1, the adversary generates *proxy prompts* that realistically simulate user behavior and produce informative traces t. Effective proxy prompts must balance realism, browsing induction, and inferential utility. (a) Prompts resemble natural-language queries real users

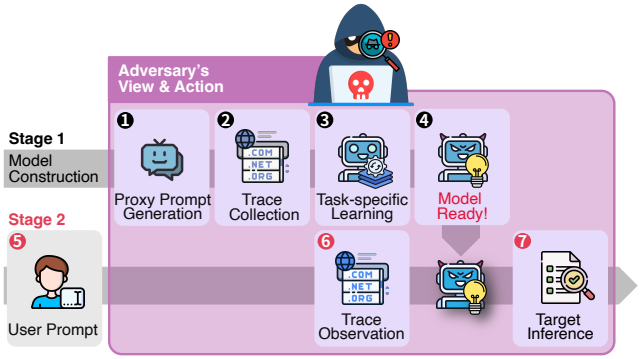

Figure 3: **Two-stage attack pipeline.**

might issue (health, education, planning). (b) They must reliably trigger multi-page browsing, rather than being satisfied by a single source (e.g., Wikipedia) or by the agent's prior knowledge; to enforce this, we prepend instructions requiring at least five distinct page visits. (c) Prompts must be sufficiently detailed to produce distinguishable traces; underspecified inputs are rewritten into instruction-complete forms to avoid trivial queries that yield little signal. (d) Prompts and resulting traces remain semantically linked, so the trace encodes sufficient signal for inference.

Prompt recovery. We draw proxy prompts from public query datasets (FEDWEB13 [18], SESSION14 [53], DD16 [54]) to build $D_{\text{PR}} = \{(WRA(p_j), p_j)\}_j$. Some prompts are underspecified to reliably induce meaningful browsing, and agents differ in how they handle such inputs. GPT-Researcher [22] internally refines vague prompts, but ChatGPT Deep Research UI asks clarifying questions, while Deep Research API, AutoGen, and Browser-Use execute inputs as-is. To align behaviors, Deep Research API prompts are rewritten into detailed variants (denoted -DR) to match the interactive interface.

Trait inference. From persona profiles, we construct $D_{\text{TI}} = \{(WRA(\{p_i^{(j)}\}_{i=1}^N), \tau_j)\}_j$, embedding selected traits either explicitly or implicitly into prompts. This mirrors real user behavior, where personal context guides information seeking (e.g., health questions tied to age or history). A persistent adversary linking sessions over time (e.g., by IP) would observe recurring patterns tied to these traits (τ_j). Executing such prompts through agents produces labeled traces that reveal stable user attributes, enabling training of trait inference models (M_{TI}). Dataset-specific sampling and generation details are described in Section 6.1.

4.2 Domain Trace Collection

To characterize how different prompts induce distinct browsing behavior, we collect domain-level metadata from agents via Playwright or built-in system logs. For each prompt p, this yields a trace $t = \langle (d_k, u_k, s_k) \rangle_{k=1}^L$, where d_k denotes accessed domain, u_k timestamp, and s_k payload size. The ordered trace captures whether an agent revisits the same domains or explores the set of sources during a single prompt execution. Although TLS application data does not expose full URLs or domain names directly, domain attribution remains feasible through DNS queries, SNI fields in TLS handshakes, or correlated DNS/HTTPS timing, all of which are observable to ISPs or other on-path entities. This matches realistic adversarial capabilities today, though we note that CDN consolidation or future deployment of DoH/DoT/ECH could reduce visibility.

Noise Filtering. A passive adversary observes all outbound requests, including primary domains, embedded resources, ads, and trackers. Because much of this traffic is incidental rather than agent-driven, an effective adversary would attempt to filter noise to isolate the domains most indicative of user intent. We apply domain-based noise filtering to remove well-known advertising and tracking services (e.g., `doubleclick.net`, `onetrust.com`, `taboola.com`). This represents the simplest plausible filtering approach and can be implemented using standard blocklists (e.g., EasyList [21]). More advanced adversaries could employ refined techniques (e.g., traffic classification or resource analysis) to further separate incidental requests from meaningful domains, which we leave to future work.

4.3 Inference Model Construction and Use

The adversary builds an inference model from proxy datasets, then applies it to new traces simulating real-user activity.

For **prompt recovery**, the task is to reconstruct a natural-language prompt that captures the functional intent of the user's original input. The adversary constructs an inference model (M_{PR}) from proxy (trace, prompt) pairs in the proxy dataset (D_{PR}), using either ICL, where a small set of examples is included in the inference prompt without parameter updates, or fine-tuning, where the same proxy pairs are used to train a direct mapping from traces to prompts (Table 2). Given a user prompt p producing trace $t = WRA(p)$ (optionally filtered to $\tilde{t}$ per Section 4.2), the resulting model M_{PR} is then applied to new traces to reconstruct real user prompts $\hat{p} = M_{\text{PR}}(\tilde{t})$.

For **trait inference**, the goal is to infer latent user attributes ($\hat{\tau}$) from browsing traces ($\tilde{t}^{1:N}$). Here, the proxy dataset (D_{TI}) is generated, consisting of proxy (trace, traits) pairs for each annotated persona profile, and the adversary adopts ICL to construct M_{TI}, since it outperformed fine-tuning in the prompt recovery task. For a new user with sessions $\{p_i\}_{i=1}^N$ and trace $t^{1:N} = WRA(\{p_i\}_{i=1}^N)$, the adversary output $\hat{\tau} = M_{\text{TI}}(\tilde{t}^{1:N})$. This enables an adversary to recover persistent attributes such as demographics, ideology, or lifestyle from observed browsing activity.

5 Ontology-aware Behavioral Leakage Score

We introduce *OBELS (Ontology-aware Behavioral Leakage Scores)*, a multi-metric scoring scheme that decomposes leak-

age into dimensions of preserved functionality, domain targets, and semantic entities, providing a structured alignment with attacker goals and real-world agent behavior.

5.1 Limitations of Existing Metrics

In AI agent contexts, leakage risk stems from *functional equivalence*: prompt recovery is risky if it leads the agent to similar sites, services, or goals, despite differences in wording. Existing metrics fall short because they assess what prompts *say*, not what they *do*.

SBERT score, computed as cosine similarity between sentence embeddings [65], captures high-level semantic alignment but suffers from paraphrastic inflation. It misleadingly assigns high scores to verbose, generic, or loosely related prompts [75, 92], overlooking task-specific behavior or intent.

LLM-as-Judge methods leverage LLMs to rate similarity through natural language instructions. While more tolerant to paraphrase and flexible, they are prompt-sensitive and often lack interpretability [39, 64], producing scalar scores without indicating which semantic elements were preserved or lost.

These shortcomings are evident in practice. For example, rephrasing "Compare Italy and Spain's digital nomad visas" as "How do work visas differ across southern Europe?" yields low cosine similarity, yet the agent navigates to similar government portals and retrieves overlapping content. From a privacy standpoint, this is a successful reconstruction, yet traditional metrics fail to capture it.

5.2 Semantic Triplet Scoring with OBELS

To address these gaps, we propose **OBELS**, a set of four complementary scores designed to capture *behavioral* alignment. Rather than collapsing leakage into a single scalar, OBELS decomposes alignment into distinct dimensions so that failures in one area (e.g., entity granularity) are not masked by strong similarity elsewhere. This decomposition yields interpretable diagnostics and aligns more directly with attacker goals and real-world agent behavior. It does so by abstracting prompts into structured triplets:

```
(intent, source_type, entity)
```

Here, `intent` denotes the high-level goal (e.g., learn, explore, analyze, compare, summarize, plan, decide, book, watch, read, evaluate); `source_type` identifies the class of information or service sought (e.g., travel, symptom, policy_area, event, visa_process, treatment_method, academic_field, cuisine, etc); and `entity` specifies the core topic (e.g., Rome, Italy, depression, face transplant, immigration, PhD, Business, Swahili dish, etc). Removing any triplet component collapses distinct browsing behaviors, whereas variations in tone or sentiment do not meaningfully alter domain traversal.

We evaluate each reconstruction across four dimensions:

- **Functional Equivalence (E_{func}):** Do the two prompts express the same high-level intent or task (e.g., finding a product, booking a service)?
- **Domain-Type Equivalence (E_{domain}):** Do they engage with the same category of services or information sources (e.g., travel agencies, health databases)?
- **Semantic Equivalence ($E_{semantic}$):** Are entities semantically aligned (e.g., "side effects of Prozac" vs. "adverse reactions to fluoxetine")?
- **Entity Granularity Tolerance (T_{entity}):** Are differences in specificity (e.g., "Italy" vs. "Rome and Venice") minor enough to preserve meaning?

Each dimension is scored from 0.0 (unrelated) to 1.0 (fully equivalent) using a standardized evaluation template applied to GPT-4, which compares triplet sets holistically and assigns fine-grained similarity scores.

Different applications may weight these components differently. For instance, entity granularity and semantic precision dominate in high-risk healthcare scenarios; domain-type alignment matters most in regulatory tasks; and intent alignment is sufficient for lower-risk consumer scenarios such as travel planning. Collapsing these dimensions into a single score would obscure such task-sensitive variation and misrepresent leakage risk.

6 Experimental Setup

We evaluate our metadata-based prompt and trait leakage attacks on held-out test prompts. The objective is to assess how well an adversary can reconstruct user prompts that induce the same downstream behavior as the originals, or infer sensitive latent traits from domain traces, thereby capturing both immediate intent leakage and longer-term profiling risks.

6.1 Datasets

We target two scenarios: prompt and trait leakage. Table 1 summarizes all datasets; full details are in Appendix C.1.

Prompt Recovery (TREC). We use three TREC datasets: FedWeb 2013 (50 topics) [52], Session 2014 (60) [53], and Dynamic Domain 2016 (53) [54], referred to as FEDWEB13, SESSION14, and DD16. For FEDWEB13 and DD16, we concatenate the `<description>` and `<narrative>` fields to match SESSION14's instruction-style and prompt specificity. To align with OpenAI's Deep Research API [58], we also generate **-DR variants** by rewriting each prompt with GPT-4.1, while scoring is always against the original prompts. For each dataset, 20 prompts are reserved for testing, with the remainder used for training and ICL examples.

TREC query text may appear in LLM pre-training corpora, but the domain-level traces we evaluate on are newly collected and do not exist in any pre-training dataset. The attack exploits these fresh behavioral traces, not memorized prompt content.

Table 1: **Traces of Web and Research Agents collected from different base datasets.**

Base Dataset	Target	# Total	# Test	Agent	Avg. # Dom	Avg. # URLs	Prompts × Sessions
FEDWEB13 [52]	Prompt	50 prompts	20	GPT-Researcher	77.5	143.0	1 prompt × 1 session
SESSION14 [53]	Prompt	60 prompts	20	GPT-Researcher	69.9	145.9	1 prompt × 1 session
DD16 [54]	Prompt	53 prompts	20	GPT-Researcher	64.8	155.8	1 prompt × 1 session
PERSONA [13]	Trait	50 personas (of 997)	3	AutoGen	144.1	173.7	3–5 prompts × 7 sessions = 21–35 queries

Trait Inference (PERSONA). We sample 50 profiles from PERSONA_subset [13] (997 total), each annotated with 32 traits. For each, we select 5 traits and use GPT-4o to generate 3–5 prompts across 7 sessions, simulating a week of browsing. Three personas are held out for ICL, and after filtering to ages 18–70, 35 remain for testing.

6.2 WRAs and Backbone LLMs

We evaluate domain-trace leakage in settings where a passive adversary can observe network-level metadata (IPs, domains, timing, payload size). To capture the traces, we use three open-source WRAs and one proprietary API agent (details in Appendix C.2):

GPT Researcher [22]. An open-source autonomous research agent for deep multi-hop investigation. We run it in `deep` mode, which produces structured research traces. GPT Researcher strategically uses multiple LLM backbones, and we evaluate both GPT-family and open-source backbones for the prompt recovery task.

Browser-Use [46]. An open-source web agent that automates browsing through a local browser. To ensure stable traces, we configure it to start from Bing and require at least five distinct page visits before summarization. Browser-Use is used for the prompt recovery task with GPT-4o as the backbone.

AutoGen [89]. An open-source web agent system we used for both prompt recovery (GPT-4o) and trait inference (Gemini 2.0 Flash), instrumented to log domains and metadata. Unlike the other agents, it produces the real packet-level traces, noisier with ads and trackers.

OpenAI Deep Research API. A proprietary research agent that serves as a commercial baseline. To fully exercise its capabilities, we provide the **-DR** prompt variant (e.g., SESSION14-DR), designed to approximate the behavior of accessing Deep Research through the ChatGPT interface. While browsing is remote and not locally visible, it serves as a benchmark for high-capability proprietary systems.

Each open-source agent is paired with both open and proprietary LLM backbones to reflect realistic local deployments. OpenAI Deep Research API serves as a proprietary baseline. Larger local LLMs (e.g., LLaMA 3.1 70B, DeepSeek-R1) are excluded because they require substantial resources and often bypass web search by relying on internal knowledge.

6.3 Evaluation Metrics

Metrics for Prompt Recovery. We evaluate prompt recovery performance using three metrics: (1) **SBERT**, computed as cosine similarity over mean-pooled Sentence-BERT embeddings [65]; (2) **LLM-as-Judge** , where an LLM rates prompt similarity based on natural language instructions; and (3) **OBELS**, our proposed behavior-aware scoring scheme to capture functional equivalence (i.e., whether the reconstructed prompt would induce similar agent behavior).

Metrics for Trait Inference. For trait inference, we adopt a type-aware scoring strategy tailored to each trait's structure:

- *Numeric traits* are evaluated using a normalized absolute difference: smaller relative errors yield higher scores, linearly decreasing toward zero.
- *Ordinal traits* are mapped to integer levels and scored by normalized distance on a 5-point scale. Closer ratings yield higher similarity scores.
- *Categorical traits* use exact match for single-token values; multi-word categories are scored using SBERT to account for semantic equivalence.
- *Free-text traits* are assessed using SBERT, which computes semantic overlap via contextual token embeddings.

This trait-type–aware evaluation avoids penalizing structured attributes with embedding-based metrics and ensures robust semantic assessment for open-ended descriptions. In contrast to prompt recovery, trait inference does not require alignment in downstream behavior; semantic similarity alone suffices to assess privacy leakage.

7 Attack Results

7.1 Prompt Recovery

Attack success is measured not by textual similarity alone, but by semantic and downstream behavior similarity, whether reconstructed prompts preserve the functional intent behind the original (see Section 6.3). The results quantify how much information about a user's input is revealed through observable browsing activity, and how the leakage rate varies across learning strategies, datasets, agents, and agents' LLM backbones. GPT-4o is used as the inference LLM unless stated

Table 2: **5-shot ICL vs. supervised fine-tuning across inference LLMs.** ICL outperforms fine-tuning on all metrics, making it the stronger approach for prompt leakage attack.

Models (Method)	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
GPT-4o (ICL)	**0.492**	**0.415**	**0.770**	**0.735**	0.520	0.640
GPT-4o (FT)	0.469	0.340	0.675	0.690	0.485	0.615
GPT-4.1 nano (ICL)	0.430	0.330	0.735	0.700	0.500	0.665
GPT-4.1 nano (FT)	0.369	0.210	0.610	0.605	0.440	0.510
Llama 3.1-8B (ICL)	0.466	0.310	0.695	0.675	0.475	0.630
Llama 3.1-8B (FT)	0.416	0.230	0.705	0.710	0.470	0.525
Qwen 2.5-32B (ICL)	0.489	0.400	0.770	0.72	**0.525**	**0.670**
Qwen 2.5-32B (FT)	0.404	0.225	0.595	0.640	0.410	0.530
Gemma 3-4B (ICL)	0.429	0.280	0.675	0.670	0.475	0.635
Gemma 3-4B (FT)	0.335	0.175	0.510	0.485	0.355	0.420

otherwise. Appendix C.3 details the ICL setup, and additional results are reported in Appendix D.

WRA vs. Human Attribution Analysis. To assess whether WRA traffic is distinguishable from human browsing, we clustered agent traces together with human sessions from SESSION14 using only timing, burstiness, and domain-transition features. WRAs consistently remain highly separable with 0.86–1.0 cluster purity across topics and 97.9% global classification accuracy. WRAs thus form a distinct, easily identifiable behavioral mode, exposing a richer and more attributable metadata footprint than humans.

ICL vs. Fine-Tuning Across Models. To motivate our choice of ICL over fine-tuning as the primary task-specific approach, we compare their performance across closed-source models (GPT-4o and GPT-4.1 nano) and open-source models (Llama 3.1 8B, Gemma 3 4B, and Qwen 2.5 32B) (Table 2). We create our final training set by combining the training samples from the prompt recovery datasets discussed in Section 6.1. For closed-source models, fine-tuning is performed through OpenAI's fine-tuning API [57] for 3 epochs with batch size 1. For open-source models, we apply LoRA [30] with rank 16 and $\alpha = 12$. Results show that ICL generally outperforms fine-tuning across nearly all metrics, though models like Llama occasionally favor fine-tuning for functional and domain equivalence. We attribute this to limited sample diversity, which likely causes overfitting in fine-tuned models.

Impact of Example Count. Table 3 compares performance across different numbers of in-context examples. 5-shot ICL provides a balanced trade-off between functional equivalence, domain alignment, and semantic similarity, while avoiding the performance drop observed at smaller or greater counts.

Impact of Agents. As described in Sections 6.1 and 6.2, we collect traces from GPT-Researcher, AutoGen, and Browser-Use (20 prompts, 5-shot ICL) and from Deep Research (5 prompts, 3-shot) using SESSION14 prompts. Table 4 shows that GPT-Researcher traces leak the most, consistent with its broader domain coverage and richer source diversity. Deep Research exhibits comparable leakage despite fewer prompts and shots, while AutoGen and Browser-Use generate learner traces that result in weaker leakage. These differences highlight how agent design and exploration strategy directly shape the degree of leakage.

Table 3: **Prompt recovery performance depending on different numbers of ICL examples.** 5-shot offers the best overall balance across evaluation metrics.

No. of Examples	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
0-shot	0.395	0.295	0.665	0.630	0.465	0.615
5-shot	**0.492**	**0.415**	**0.770**	**0.735**	0.520	0.640
8-shot	0.490	0.400	0.760	0.730	**0.540**	0.610
12-shot	0.504	0.405	0.750	0.710	0.530	0.640
15-shot	0.487	0.385	0.770	0.685	0.525	**0.645**

Table 4: **Prompt recovery performance on SESSION14 across WFAs.** All agents are evaluated on 20 prompts with 5-shot ICL, except Deep Research* (5 prompts, 3-shot). GPT-Researcher traces leak the most, while Deep Research shows comparable leakage despite fewer shots.

Agent	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
AutoGen	0.356	0.225	0.565	0.625	0.400	0.495
Browser-Use	0.368	0.240	0.540	0.590	0.390	0.485
GPT-Researcher	0.492	**0.415**	**0.770**	**0.735**	**0.520**	**0.640**
Deep Research*	**0.574**	0.360	0.680	0.720	0.480	0.640

Impact of Agent Backbone: Proprietary vs. Open-Source. Using GPT-Researcher, we evaluated traces across datasets with either GPT-4 or local open-source backbones (DeepSeek-V3, Mistral family). As shown in Table 5, GPT-4 traces yield higher recovery scores, reflecting stronger reasoning capacity and broader exploration. Local models leak less overall, but still provide exploitable cues. For the remainder of our ICL experiments, we use GPT-Researcher with a GPT-4 backbone on SESSION14, as this setup offers both representativeness and clearer leakage signals for analysis.

Impact of Inference LLM. Table 6 compares inference engines for ICL-based prompt recovery, showing that model choice strongly affects reconstruction quality. Claude-opus-4-1 and Gemini-2.5-pro achieve the highest scores, while GPT-5, GPT-5-mini, and GPT-4o remain competitive. Larger models leak more information but incur higher cost and latency, especially in the Gemini, Claude, and GPT-5 families. Balancing accuracy, cost, and latency, we identify GPT-4o as the most practical option, offering strong recovery performance with lower overhead than top-scoring but slower systems.

Table 5: **Prompt recovery on trace by GPT-Researcher with different backbones across datasets.** Local backbones leak less than GPT-4 traces but remain informative.

Model	Dataset	SBERT	LLM-Judge	OBELS			
				E_{func}	E_{dom}	E_{sem}	T_{ent}
GPT4	SESSION	0.492	0.415	0.770	0.735	0.520	0.640
Local	SESSION	0.512	0.355	0.735	0.745	0.520	0.630
GPT4	FEDWEB	0.505	0.305	0.755	0.735	0.540	0.695
Local	FEDWEB	0.493	0.310	0.750	0.745	0.515	0.645
GPT4	DD	0.453	0.190	0.585	0.615	0.430	0.545
Local	DD	0.445	0.180	0.550	0.635	0.430	0.515

Table 6: **Prompt recovery with different ICL inference models.** Proprietary models yield the highest scores.

Models	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
Claude-opus-4-1	**0.544**	**0.445**	0.765	**0.785**	**0.570**	**0.670**
Gemini-2.5-pro	0.506	0.420	0.710	0.705	0.505	0.645
GPT-5	0.470	0.240	0.710	0.715	0.535	0.670
GPT-5-mini	0.479	0.315	0.710	0.685	0.500	0.635
GPT-4o	0.492	0.415	**0.770**	0.735	0.520	0.640
GPT-4.1 nano	0.430	0.330	0.735	0.700	0.500	0.665
Llama 3.1 8B	0.466	0.310	0.695	0.675	0.475	0.630
Qwen 2.5-32B	0.489	0.400	0.770	0.72	0.525	0.670
Gemma 3-4B	0.429	0.280	0.675	0.670	0.475	0.635

7.2 Trait Inference

We assess how much and accurately an adversary can recover latent attributes of the user from a week-long browsing trace (see Section 6.3). The results quantify trait exposure using *inference accuracy* (similarity score), where higher values mean more reliable adversarial inference. Gemini 2.5 pro is used as the inference LLM.

Figure 4 ranks the fifteen most exposed traits, showing that exposure varies by attribute. Health insurance (0.98), veteran status (0.90), employment status (0.88), and household language (0.86) are inferred with near-perfect accuracy, with employment status nearly as reliable as binary traits despite being multi-class. A second tier (marital status, sex, race, religion, household type) scores 0.73–0.77, while political views, citizenship, education, family presence/age, place of birth, and income range from 0.66–0.70, still well above random. Behavioral and psychographic traits do not appear in the top 15, consistent with their lower exposure in Figure 6. Overall, the results suggest that demographic and occupational traits are most vulnerable to leakage, while lifestyle and personality attributes remain less exposed but at nontrivial levels.

Figure 5 compares similarity scores for selected and unselected traits. Selected traits typically exceed 0.7, with some personas (012, 024, 037, 045) surpassing 0.8, indicating consistent high-risk leakage [10]. Unselected traits generally fall below but often remain above 0.5, showing moderate exposure even when traits are not explicitly embedded. In a few cases (007, 025, 041), unselected traits approach or exceed selected ones, confirming leakage extends beyond explicitly mentioned attributes.

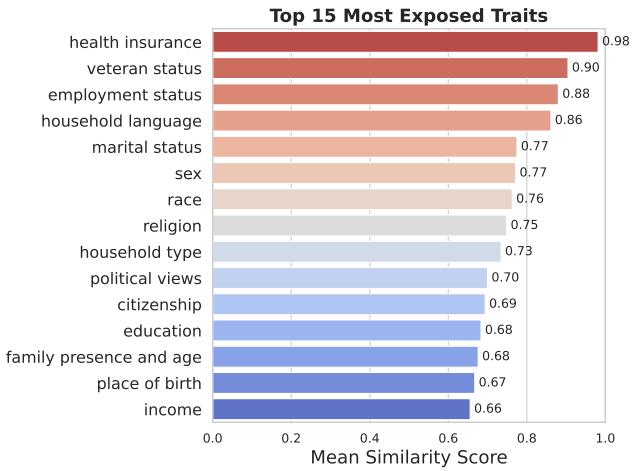

Figure 4: **Top 15 traits with highest exposure risk.** Inference risk is measured by mean similarity scores across all personas and sessions. Higher scores indicate stronger inference.

Figure 6 compares model confidence and accuracy across categories. As shown in the left plot, occupational and demographic traits are assigned very high confidence (0.8–1.0), while the other two peak in the mid-confidence range (0.6–0.8). The accuracy distributions (right plot) further distinguish the categories: occupational traits average 0.70 but with a lower median (0.53), psychographic traits are inferred with moderate accuracy (mean 0.56, median 0.46) but show extreme variability (spread 0–1). Behavioral traits are uniformly moderate to weak (mean 0.46, median 0.45) with a narrowest spread of 0.25 to 0.61, and demographic traits show the highest median (0.90), but occasional errors lower the mean (0.44). Trait categories differ not only in accuracy but also in distribution: demographic traits are most reliably inferred, occupational traits are less consistent but often highly accurate, psychographic traits are highly variable, and behavioral traits are the least exposed.

Different Number of Sessions. As part of our ablation study, Figure 7 shows the distribution of traits per persona with similarity scores above 0.7 under three vs. seven sessions. With three sessions (yellow), most personas reveal 6–10 traits with a long tail of higher counts. With seven sessions (purple), the distribution shifts rightward and concentrates around 12–14 traits, with some reaching 19. This demonstrates that additional sessions, and thus more domain traces, enable inference of more traits at high accuracy.

Table 7 confirms consistent gains across categories, with psychographic (+16.7%) and demographic (+15.4%) improving most, followed by behavioral (+12.2%) and occupational (+7.4%) traits. Overall, longer observation windows

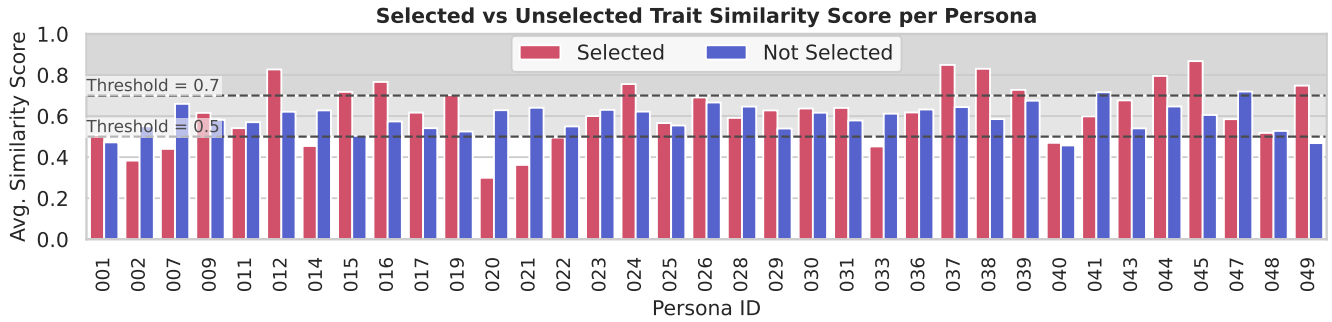

Figure 5: **Average similarity scores per persona for selected (red) and unselected (blue) traits.** Dashed lines at 0.5 and 0.7 indicate moderate and high-risk leakage thresholds. Many personas have selected traits exceeding 0.7, while unselected traits frequently remain above 0.5, reflecting moderate but non-negligible exposure.

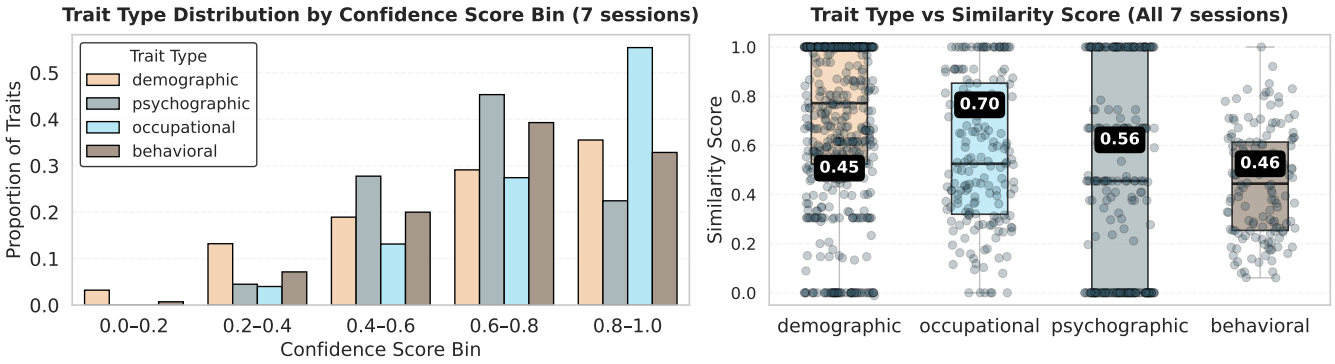

Figure 6: **Left:** Proportion of inferred traits in each confidence bin. Occupational and demographic traits are most often assigned high confidence (0.8–1.0), while the other two peak in middle ranges (0.6–0.8). **Right:** Similarity scores by trait category. Boxes show the interquartile range, black lines, squared numbers, and dots represent medians, means, and all data points, respectively.

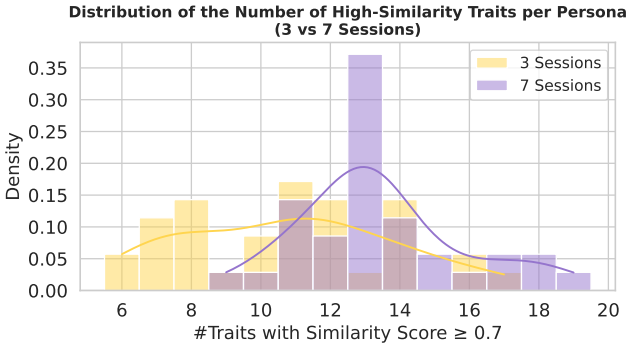

Figure 7: Distribution of traits per persona with similarity $\geq$ 0.7 under 3- and 7-session settings. The 7-session case shifts rightward, indicating more traits inferred at high similarity.

strengthen inference, increasing both the number of high-similarity traits and average accuracy.

Overall, our results show that metadata leakage enables both immediate and long-term risks. First, prompts can be reliably recovered regardless of whether traces are produced by proprietary or local LLM backbones, or by agents with varying exploration strategies. For trait inference, exposure levels differ across categories, but even traits not explicitly embedded in prompts can still be inferred at moderate to high accuracy, as shown by the recovery of unselected traits. Together, these findings demonstrate that domain traces alone reveal sensitive user intent and identity signals, underscoring metadata leakage as a significant privacy threat.

Table 7: **Average similarity scores across trait categories.** Δ shows the improvement of full sessions over three sessions.

Trait Category	3 Sessions	7 Sessions	Δ (%)
Demographic	0.39	0.45	+15.4%
Occupational	0.68	0.73	+7.4%
Psychographic	0.48	0.56	+16.7%
Behavioral	0.41	0.46	+12.2%

8 Defense and Realistic Deployment

Motivation. If a task requires visiting a unique domain, the defense should hide that interaction via VPNs or randomized

background queries. By contrast, if the information can be obtained without distinctive traces, blocking is preferable; for example, by querying an LLM directly or gathering information from Wikipedia. We thus identify two conceptual mitigations for metadata-based inference attacks: (1) **hiding traces**, where sensitive activity must occur but is camouflaged with plausible noise; and (2) **blocking traces**, where the activity is avoided by relying on alternative, less revealing sources.

8.1 Hiding via Decoy Prompts

Inspired by TrackMeNot [51], for each real prompt the system generates and executes additional decoy prompts that remain within the same topical space but differ in framing, intent, or implied traits. Because OBELS evaluates whether reconstructions preserve intent, domain type, and entity, breaking this alignment is critical; otherwise, an adversary can still recover the user's functional behavior despite wording changes. For instance, a prompt about "recognizing depression and seeking medical help for a friend" can be paired with a decoy like "compare the effectiveness of yoga versus meditation for improving sleep quality", which shifts both domain and entity, lowering OBELS score and obscuring the true goal. This ambiguity in observed traffic undermines an adversary's ability to infer the user's true objective, attributes, or target entity. We explore a lightweight approach using LLM-based decoy generation and a trait-aware variant that leverages virtual personas to introduce long-term behavioral divergence.

Mechanism. The defense operates in three stages. (1) **Decoy prompt generation**: an LLM produces multiple decoys that remain within the same topical space but differ in context, intent, or entity granularity, thereby obscuring both high-level intent and fine-grained attributes such as geographic focus or institutional target (e.g., a prompt on "Swahili food" may be paired with decoys on "Italian cuisine" or "modern American dishes"). (2) **Trait-conflicting guidance** (optional): decoy generation can be steered by a virtual persona, with the system maintaining a rolling estimate of the user's traits via lightweight keyword heuristics and selecting a persona that diverges on key dimensions (e.g., ideology or religion). These divergent decoys introduce consistent but misleading patterns, weakening long-term profiling. (3) **Concurrent execution**: real and decoy prompts are issued in parallel; while only the real output is returned to the user, all queries leave observable traces, blending plausible yet misleading activity into the traffic and complicating adversarial inference of the user's true intent or identity.

Impact of Defense on Prompt Leakage Attack. Table 8 summarizes the prompt recovery performance under defense applied. The defense injects a varying number of decoy prompts, each contributing its own set of visited domains, so that the agent runs for both the original and decoy prompts simultaneously. Original domains are randomly interleaved within these sets while preserving their internal order, as shown in rows 2–4 of the table. To model a stronger defender, we also consider a condition where all domains, both original and decoy, are fully shuffled (rows 5–7).

Table 8: **Effect of decoy prompts and domain shuffling on prompt leakage attacks.**

Defense	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
Without Defense	0.492	0.415	0.770	0.735	0.520	0.640
1 Decoy	0.390	0.190	0.610	0.615	0.420	0.545
3 Decoys	0.376	0.210	0.645	0.615	0.420	0.555
5 Decoys	**0.368**	**0.175**	**0.565**	**0.590**	**0.395**	**0.500**
1 Decoys + shuffle	0.455	0.285	0.725	0.715	0.510	0.610
3 Decoys + shuffle	0.415	0.235	0.685	0.645	0.455	0.610
5 Decoys + shuffle	0.403	0.220	0.650	0.695	0.465	0.570

Table 9: **Median inference accuracy across trait categories.**

Trait Category	No Defense	w. Defense	Δ (%)
Demographic	0.4440	0.4080	−8.1%
Occupational	0.8975	0.6790	−24.3%
Psychographic	0.5250	0.5030	−4.2%
Behavioral	0.4550	0.4550	0.0%

The results show a consistent reduction in inference accuracy once decoys are added. With only one decoy prompt, SBERT drops by 0.10 and LLM-Judge by 0.23 relative to the baseline. More decoys generally strengthen the defense, with the largest effect observed at five decoys. When domains are shuffled, attack performance recovers somewhat for semantic and domain-level metrics, but still remains below baseline.

Impact of Defense on Trait Inference. We evaluate the defense by re-running the trait inference attack with decoy prompts injected. In this setting, each real prompt triggers a concurrent decoy prompt from a conflicting virtual persona, selected based on the defender-side trait estimate. Table 9 reports median similarity scores between inferred and ground-truth traits across four categories. The defense yields the largest median reduction for occupational traits (−24.3%), followed by demographic traits (−8.1%). Psychographic traits drop slightly (−4.2%), while behavioral traits remain unchanged. Additional results are provided in Appendix E.

8.2 Blocking via Alternative Sources

A complementary defense is to prevent sensitive traces from arising in the first place. When a user's objective can be met without contacting uniquely identifying domains, the agent can instead draw from large multipurpose repositories (e.g., Wikipedia, StackExchange, Reddit) or from the LLM's internal knowledge. By retrieving information from resources that serve diverse intents or by leveraging the model's internal knowledge, the resulting traffic no longer maps cleanly to a

Table 10: **Prompt recovery under different trace visibility.** URL-level traces leak the most; adding timing metadata shows no consistent benefit.

Trace Visibility	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
Domains	0.492	0.415	0.770	0.735	0.520	0.640
Domains + Timing	0.486	0.455	0.725	0.695	0.525	0.640
URLs (100%)	**0.624**	**0.600**	0.810	**0.805**	0.620	**0.695**
Partial URLs (80%)	0.618	0.590	0.795	0.795	**0.635**	0.680
Partial URLs (60%)	0.622	0.590	**0.820**	0.755	0.585	0.690

Table 11: **Leakage vs. utility under varying visibility.** Reducing visibility lowers prompt recovery accuracy but leaves the report quality nearly unchanged.

Visibility	Utility	SBERT	LLM-Judge	OBELS			
				E_{func}	E_{dom}	E_{sem}	T_{ent}
100%	8.55	0.492	0.415	0.770	0.735	0.520	0.640
80%	8.72	0.491	0.415	0.750	0.725	0.515	0.645
60%	8.54	0.489	0.355	0.765	0.730	0.520	0.650
40%	8.50	0.453	0.350	0.765	0.700	0.515	0.650
20%	8.60	0.438	0.300	0.725	0.695	0.475	0.615
10%	8.54	0.375	0.225	0.610	0.555	0.395	0.485
5%	8.32	0.319	0.185	0.530	0.455	0.335	0.395
2%	7.65	0.238	0.095	0.345	0.410	0.275	0.315

specific user goal.

From the adversary's perspective, this strategy effectively reduces visibility by collapsing domain diversity, yielding traces similar to those observed by weaker observers that see limited or non-discriminative domain information. This "blocking" approach therefore both mitigates leakage and illustrates how reduced observability weakens inference, reducing the adversary's ability to associate domain visits with private attributes.

Table 10 and Table 11 together show that visibility of browsing traces strongly shapes leakage but has little effect on utility. Utility is how much the WRA's output helps a human reader reach their research goals. We measure it with an LLM using the prompt template. URL-level traces leak substantially more than domain-only traces, making prompt inference easier and more accurate. Reducing visibility (e.g., observing only 80% or 60% of domains) lowers leakage but still exposes useful information. Adding timing metadata to domains yields no consistent gains, suggesting that domain identity is more informative than temporal order. In contrast, the utility of the generated reports remains relatively stable across visibility levels down to 10%. However, beyond this threshold, utility begins to drop, revealing diminishing returns from additional domain coverage. This suggests that *many extra domains contribute little to report quality while disproportionately increasing exposure to adversaries*. Thus, WRA behaviors that aggressively visit many domains are especially problematic: they create substantial leakage risks while providing only marginal gains in utility.

9 Discussion and Conclusion

Privacy and Security Implications. Our results show that metadata leakage is an inherent risk for locally deployed web and research agents. We find that design choices—including domain exploration breadth, backbone model capacity, and orchestration logic—directly affect the amount of information exposed. Substituting locally deployable models does not eliminate leakage: domain-level traffic is structurally unavoidable. This shifts the attack vector from *content* to *behavior*. Whereas prompt injection and data exfiltration could be mitigated through isolation [81], network-level traces cannot be trivially suppressed. At scale, such leakage enables profiling by advertisers, ISPs, or state actors, raising unique questions about how to balance agent utility with user privacy.

Defense Effectiveness and Limitations. Our study has several limitations. We rely on proxy datasets and synthetic personas with limited session counts; longer-term aggregation or multimodal attacks may amplify risks. Defense evaluation is confined to decoys and blocking, and cost/latency trade-offs are only partially measured. Nonetheless, our results show that decoy prompts and trait-conflicting personas reduce both prompt recovery and trait inference accuracy, while blocking strategies help when tasks can rely on broad, non-unique sources (e.g., Wikipedia) without sacrificing utility. Yet defenses remain partial: adversaries still recover useful signals, and decoys introduce overhead in latency, bandwidth, and behavioral realism. As with prior obfuscation systems, mitigation is meaningful but incomplete; full network-level protections (e.g., VPNs or anonymity systems for agents [62]) remain the only robust safeguard.

Future Directions and Open Challenges. Future work should explore hybrid defenses that combine obfuscation, blocking, and timing perturbations; study long-term aggregation attacks across extended user histories; and integrate formal privacy guarantees such as differential privacy or traffic-analysis resistance directly into agent design. Multi-modal agents introduce additional risks, for example, as passed multi-modal content could reveal even more information [70]. Overall, progress requires viewing AI agents and the web as a coupled ecosystem rather than orthogonal components.

Conclusion. We show prompt and trait leakage from network traces of local WRAs. Even with defenses, residual exposure persists because traces are structurally generated and behaviorally rich. Our study demonstrates both the feasibility of inference attacks and the incompleteness of current defenses, underscoring the need for systemic, privacy-aware agent design. Until such frameworks are developed, strong network-level protections remain the only reliable safeguard.

Acknowledgments

The work was partially supported by Schmidt Sciences SAFE-AI program and by the NSF grants 2333965 and 2131910.

A Ethical Considerations

Our research examines how WRAs can inadvertently leak sensitive information through metadata such as domain access patterns, timing, and interaction traces. While our goal is to strengthen privacy protections for agent-based systems, we recognize the potential for harmful use if such inference techniques were deployed maliciously.

Stakeholders. We identify four stakeholder groups affected by this research: (1) End-users of WRAs, who may have personal traits, goals, or intentions inferred without their awareness. This risk is not uniform; individuals in politically sensitive contexts, marginalized communities, or high-surveillance environments may face disproportionately higher harm. (2) Developers and framework maintainers, who may need to modify or harden system architectures, logging defaults, and data retention practices in response to identified risks. (3) Organizations deploying agents (e.g., educational platforms, customer service systems), which may unknowingly violate privacy expectations or regulatory obligations if leakage is not addressed. (4) The broader public, which has a collective interest in ensuring that emerging agent ecosystems evolve in privacy-preserving and autonomy-respecting directions.

Mitigating Potential Harm. To reduce risks to end-users, we designed our study so that no real user data were collected or analyzed; all experiments use public benchmark datasets (FEDWEB13, SESSION14, DD16) and synthetic personas, ensuring that no identifiable individuals can be profiled or re-identified. To support developers and framework maintainers, we present concrete mitigation strategies, such as trace obfuscation and domain suppression, that can be incorporated into WRA architectures to reduce metadata leakage. For organizations that deploy agents in educational, corporate, or customer-service environments, we do not release any tools for live traffic monitoring, and the accompanying code is limited to offline, dataset-based replay experiments, preventing misuse in operational settings. Finally, by identifying these risks early and providing actionable defenses, the work serves broader public interests by promoting privacy-preserving WRA design, rather than end-user surveillance or behavioral profiling as agents become widely deployed.

Researcher Well-being and Responsibilities. Our research did not involve direct interaction with online communities, covert participation, or exposure to disturbing or sensitive content. However, we continuously discussed the dual-use concerns throughout the project and made design choices aimed at minimizing the potential for misuse, for example, limiting our evaluation to public datasets and not designing tools that can be directly applied to live user traffic.

Societal Impact and Justification. As WRAs become integrated into productivity tools, classrooms, workplaces, and personal browsing workflows, metadata-based inference risks could scale rapidly and invisibly. By identifying and characterizing this attack vector under controlled conditions, we enable developers, regulators, and practitioners to recognize and address these vulnerabilities early, before they are exploited in real-world deployments. By focusing on public data, transparent methodology, and concrete mitigation guidance, we ensure that the research advances privacy-preserving agent design while minimizing the likelihood of harmful use.

B Open Science

Implementations, evaluation code, prompts, and the synthetic datasets derived from our study will be made publicly available. Code and prompts available at: https://github.com/umass-aisec/wra.

References

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *CCS*, 2014.

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

[3] AMD. GAIA: Generative AI is awesome (open-source project). https://github.com/amd/gaia, 2025.

[4] Mika D Ayenson, Dietrich James Wambach, Ashkan Soltani, Nathan Good, and Chris Jay Hoofnagle. Flash cookies and privacy ii: Now with HTML5 and etag respawning. *Available at SSRN 1898390*, 2011.

[5] Daniel Ayzenshteyn, Roy Weiss, and Yisroel Mirsky. Cloak, honey, trap: Proactive defenses against LLM agents. In *USENIX Security*, 2025.

[6] Danny Azucar, Davide Marengo, and Michele Settanni. Predicting the big 5 personality traits from digital footprints on social media: A meta-analysis. *Personality and individual differences*, 124:150–159, 2018.

[7] Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. AirGapAgent: Protecting privacy-conscious conversational agents. In *CCS*, 2024.

[8] Sarah Bird, Ilana Segall, and Martin Lopatka. Replication: whywe still can't browse in peace: on the uniqueness and reidentifiability of web browsing histories. In *SOUPS*, 2020.

[9] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *CCS*, 2012.

[10] Tristan JB Cann, Ben Dennes, Travis Coan, Saffron O'Neill, and Hywel TP Williams. Using semantic similarity to measure the echo of strategic communications. *EPJ Data Science*, 14(1):20, 2025.

[11] Nicholas Carlini and Milad Nasr. Remote timing attacks on efficient language model inference. *arXiv:2410.17175*, 2024.

[12] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *USENIX Security*, 2021.

[13] Louis Castricato, Nathan Lile, Rafael Rafailov, Jan-Philipp Fränken, and Chelsea Finn. PERSONA: A reproducible testbed for pluralistic alignment. In *COLING*, 2025.

[14] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[15] Sijia Chen, Yibo Wang, Yi-Feng Wu, Qingguo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Lijun Zhang. Advancing tool-augmented large language models: Integrating insights from errors in inference trees. *NeurIPS*, 2024.

[16] Jeffrey Yang Fan Chiang, Seungjae Lee, Jia-Bin Huang, Furong Huang, and Yizheng Chen. Why are web AI agents more vulnerable than standalone llms? a security analysis. In *ICLR Workshop*, 2025.

[17] Kyle Crichton, Lorrie Faith Cranor, and Nicolas Christin. Rethinking fingerprinting: An assessment of behavior-based methods at scale and implications for web tracking. In *PETS*, 2025.

[18] Thomas Demeester, Dolf Trieschnigg, Dong Nguyen, and Djoerd Hiemstra. Overview of the TREC 2013 federated web search track. In *TREC*, 2013.

[19] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *SSYM*, 2004.

[20] Santiago (Sal) Díaz, Christoph Kern, and Kara Olive. Google's approach for secure AI agents. Technical report, Google, 2025.

[21] EasyList Community. EasyList: The Primary Filter List Used by Adblockers. https://github.com/easylist/easylist, 2025. GitHub repository.

[22] Assaf Elovic. GPT researcher. Code repository: https://github.com/assafelovic/gpt-researcher.

[23] Xiang Gao and Kamalika Das. Customizing language model responses with contrastive in-context learning. In *AAAI*, 2024.

[24] Google. Gemini Deep Research: Your personal AI research assistant. https://gemini.google/overview/deep-research/, 2025. Accessed: 2025-07-16.

[25] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *ICML*, 2020.

[26] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *USENIX Security*, 2016.

[27] Andrew Hintz. Fingerprinting websites using traffic analysis. In *International workshop on privacy enhancing technologies*, pages 171–178. Springer, 2002.

[28] Nguyen Phong Hoang, Arian Akhavan Niaki, Nikita Borisov, Phillipa Gill, and Michalis Polychronakis. Assessing the privacy benefits of domain name encryption. In *AsiaCCS*, 2020.

[29] Dirk Hovy. Demographic factors improve classification performance. In *ACL-IJCNLP*, 2015.

[30] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-rank adaptation of large language models. *ICLR*, 2022.

[31] Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, Jianye Hao, et al. Deep research agents: A systematic examination and roadmap. *arXiv:2506.18096*, 2025.

[32] Bo Hui, Haolin Yuan, Neil Gong, Philippe Burlina, and Yinzhi Cao. PLeak: Prompt leaking attacks against large language model applications. In *CCS*, 2024.

[33] Glory Jain and Kevin Alwell. Introduction to deep research in the openai api. https://cookbook.openai.com/examples/deep_research_api/introduction_to_deep_research_api, 2025.

[34] Rosie Jones, Ravi Kumar, Bo Pang, and Andrew Tomkins. "i know what you did last summer" query logs and user privacy. In *CIKM*, 2007.

[35] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *ESORICS*, 2016.

[36] Melanie Kellar, Carolyn Watters, and Michael Shepherd. A field study characterizing web-based information-seeking tasks. *Journal of the American Society for information science and technology*, 58(7):999–1018, 2007.

[37] Hanna Kim, Minkyoo Song, Seung Ho Na, Seungwon Shin, and Kimin Lee. When LLMs go online: The emerging threat of web-enabled LLMs. In *USENIX Security*, 2025.

[38] LangChain AI. Local Deep Researcher: A fully local web research assistant. https://github.com/langchain-ai/local-deep-researcher, 2025.

[39] Yoonjoo Lee, Kihoon Son, Tae Soo Kim, Jisu Kim, John Joon Young Chung, Eytan Adar, and Juho Kim. One vs. many: Comprehending accurate information from multiple erroneous and inconsistent ai generations. In *FAccT*, 2024.

[40] Ada Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. Internet jones and the raiders of the lost trackers: An archaeological study of web tracking from 1996 to 2016. In *USENIX Security*, 2016.

[41] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *NeurIPS*, 2020.

[42] Yupei Liu, Yuqi Jia, Jinyuan Jia, and Neil Zhenqiang Gong. Evaluating large language model based personal information extraction and countermeasures. In *USENIX Security*, 2025.

[43] Liv McMahon. Hundreds of thousands of grok chats exposed in google results. *BBC News*, August 21 2025.

[44] Mistral AI. Le chat dives deep. Mistral AI blog, Jul 2025. Announcing "Le Chat," a powerful deep reasoning model by Mistral AI.

[45] Mykyta Mudryi, Markiyan Chaklosh, and Grzegorz Wójcik. The hidden dangers of browsing AI agents. *arXiv:2505.13076*, 2025.

[46] Magnus Müller and Gregor Žunič. Browser use: Enable ai to control your browser. https://github.com/browser-use/browser-use, 2025.

[47] Farid M Naini, Jayakrishnan Unnikrishnan, Patrick Thiran, and Martin Vetterli. Where you are is who you are: User identification by matching statistics. *TIFS*, 2015.

[48] Arvind Narayanan, Hristo Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. On the feasibility of internet-scale author identification. In *S&P*, 2012.

[49] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *S&P*, 2008.

[50] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *S&P*, 2013.

[51] Helen Nissenbaum and Howe Daniel. Trackmenot: Resisting surveillance in web search. *Lessons from the Identity Trail*, 2009.

[52] NIST. TREC 2013 federated web track: 50 topics (fedweb13). XML file 'fedweb13_50topics.xml', 2013. Available from the TREC data repository, NIST.

[53] NIST. TREC 2014 session track data. Dataset files (topics, session→topic mapping, relevance judgments), 2014. Available from the TREC data repository, NIST.

[54] NIST. TREC 2016 dynamic domain track data. Dataset files, overview materials, 2016. Available from the TREC data repository, NIST.

[55] Marcos Oliveira, Junran Yang, Daniel Griffiths, Denis Bonnay, and Juhi Kulshrestha. Browsing behavior exposes identities on the web. *arXiv:2312.15489*, 2023.

[56] Ollama Team. Ollama: Run large language models locally. https://ollama.com, 2025.

[57] OpenAI. Fine-tuning now available for GPT-4o — openai.com. https://openai.com/index/gpt-4o-fine-tuning/, 2025.

[58] OpenAI. Introducing Deep Research. https://openai.com/index/introducing-deep-research/, 2025.

[59] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In *WPES*, 2011.

[60] Dario Pasquini, Evgenios M Kornaropoulos, and Giuseppe Ateniese. LLMmap: Fingerprinting for large language models. In *USENIX Security*, 2025.

[61] Perplexity AI. Introducing *Deep Research*. Perplexity AI blog, Feb 2025. "Deep Research performs dozens of searches, reads hundreds of sources, and delivers comprehensive reports".

[62] Dzung Pham, Jade Sheffey, Chau Minh Pham, and Amir Houmansadr. ProxyGPT: Enabling anonymous queries in AI chatbots with (un)trustworthy browser proxies. *arXiv:2407.08792*, 2024.

[63] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *ICLR*, 2024.

[64] Amirhossein Razavi, Mina Soltangheis, Negar Arabzadeh, Sara Salamat, Morteza Zihayat, and Ebrahim Bagheri. Benchmarking prompt sensitivity in large language models. In *ECIR*, 2025.

[65] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *EMNLP-IJCNLP*, 2019.

[66] Daniel E Rose and Danny Levinson. Understanding user goals in web search. In *WWW*, 2004.

[67] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023.

[68] Samuel Schmidgall and Michael Moor. AgentRxiv: Towards collaborative autonomous research. *arXiv:2503.18102*, 2025.

[69] Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using LLM agents as research assistants. *arXiv:2501.04227*, 2025.

[70] Roei Schuster, Vitaly Shmatikov, and Eran Tromer. Beauty and the burst: Remote identification of encrypted video streams. In *USENIX Security*, 2017.

[71] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, et al. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, 2013.

[72] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. *NeurIPS*, 2023.

[73] Sandra Siby, Marc Juarez, Claudia Diaz, Narseo Vallina-Rodriguez, and Carmela Troncoso. Encrypted DNS–> privacy? a traffic analysis perspective. In *NDSS*, 2020.

[74] Significant-Gravitas. AutoGPT: Build, deploy, and run AI agents. https://github.com/Significant-Gravitas/AutoGPT.

[75] Harald Steck, Chaitanya Ekanadham, and Nathan Kallus. Is cosine-similarity of embeddings really about similarity? In *WWW*, 2024.

[76] Chris Stokel-Walker. Exclusive: Google is indexing chatgpt conversations, potentially exposing sensitive user data. *Fast Company*, July 30 2025.

[77] Jessica Su, Ansh Shukla, Sharad Goel, and Arvind Narayanan. De-anonymizing web browsing data with social networks. In *WWW*, 2017.

[78] Satoru Sunahara, Yong Jin, Katsuyoshi Iida, and Yoshiaki Takai. A framework for institutional privacy considered full dns over https architecture. *IEEE Access*, 2025.

[79] Tavily. Tavily: The web access layer for ai agents. https://www.tavily.com/, 2025.

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

[81] Lillian Tsai and Eugene Bagdasarian. Contextual agent security: A policy for every purpose. In *HotOS*, 2025.

[82] Bo Wang, Weiyi He, Pengfei He, Shenglai Zeng, Zhen Xiang, Yue Xing, and Jiliang Tang. Unveiling privacy risks in LLM agent memory. In *ACL*, 2025.

[83] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. In *NeurIPS Workshops*, 2025.

[84] Mona Wang, Pellaeon Lin, Jeffrey Knockel, Will Greenberg, Jonathan Mayer, and Prateek Mittal. What WeChat knows: Pervasive first-party tracking in a billion-user super-app ecosystem. *PETS*, 2025.

[85] Zora Zhiruo Wang, Yijia Shao, Omar Shaikh, Daniel Fried, Graham Neubig, and Diyi Yang. How do ai agents do human work? comparing ai and human workflows across diverse occupations. *arXiv preprint arXiv:2510.22780*, 2025.

[86] Jiankun Wei, Abdulrahman Abdulrazzag, Tianchen Zhang, Adel Muursepp, and Gururaj Saileshwar. Privacy risks of speculative decoding in large language models. *arXiv:2411.01076*, 2024.

[87] Roy Weiss, Daniel Ayzenshteyn, and Yisroel Mirsky. What was your prompt? a remote keylogging attack on AI assistants. In *USENIX Security*, 2024.

[88] Charles V Wright, Scott E Coull, and Fabian Monrose. Traffic morphing: An efficient defense against statistical traffic analysis. In *NDSS*, 2009.

[89] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *COLM*, 2024.

[90] xAI. Grok 3 beta — the age of reasoning agents. xAI blog, Feb 2025. Announcing Grok 3, a reasoning-optimized version of the Grok chatbot.

[91] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

[92] Kisung You. Semantics at an angle: When cosine similarity works until it doesn't. *arXiv:2504.16318*, 2025.

[93] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *ACL*, 2024.

[94] Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. Effective prompt extraction from language models. In *COLM*, 2024.

[95] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv:2504.03160*, 2025.

C Experimental Setup Details

C.1 Dataset Details

Prompt Recovery. For all TREC datasets used in the prompt recovery task, we create **-DR** variants by inputting each original prompt with `suggested_rewriting_prompt` [33], to GPT-4.1. The rewritten prompts are used for the Deep Research API runs to match its prompt handling, and all evaluations are scored against the original, unmodified prompts.

TREC FedWeb 2013 Topics contains 50 topics originally designed to evaluate federated web search. We concatenate the `<description>` and `<narrative>` fields to form a single prompt, and rewritten with GPT-4.1 (FEDWEB13-DR).

TREC Session 2014 Topics contains 60 realistic user topic descriptions spanning domains such as travel, health, and education. We use the original `description` fields as user prompts and their rewritten counterparts (SESSION14-DR).

Table 12: Few-shot ICL configuration for each attack task.

Task	ICL Shots	Output Format	Prompt Details
Prompt Recovery	5	Natural language query	Trace → prompt
Trait Inference	3	Structured list	Trace → trait list

TREC Dynamic Domain 2016 Truth Data provides 53 topics focused on interactive information retrieval in evolving domains such as medicine and cybersecurity. We concatenate the `<description>` and `<narrative>` fields and rewrite them with GPT-4.1 (DD16-DR).

For each dataset, 20 prompts are reserved for evaluation, with the remainder used for training and ICL examples.

Trait Inference Personas. We use **SynthLabsAI/PERSONA_subset** to simulate benign users who use a web agent over time. The dataset includes 997 synthetic user profiles annotated with 32 traits in four categories (demographic, occupational, psychographic, and behavioral). We adopt a type-aware scoring for each trait for evaluation:

- *Numeric traits* (e.g., age, income) are evaluated using a normalized absolute difference: smaller relative errors yield higher scores, linearly decreasing toward zero. Age differences are scaled by 30, and income by 200K.
- *Ordinal traits* (e.g., big five scores) are mapped to integer levels and scored by normalized distance on a 5-point scale. Closer ratings yield higher similarity scores.
- *Categorical traits* (e.g., gender, race) use exact match for single-token values; multi-word categories are scored using SBERT score to account for semantic equivalence.
- *Free-text traits* (e.g., job description, personal time) are assessed using the SBERT score, which computes semantic overlap via contextual token embeddings.

In our experiments, we sample 50 personas. For each, including those reserved for ICL examples, we randomly select 5 traits and use GPT-4o ($T = 0.7$) to generate 21–35 trait-revealing prompts that explicitly or implicitly embed the traits. The produced traces serve as the input for trait inference.

C.2 Agent and LLM Backbone Choice

When browsing is executed locally, encrypted outbound traffic remains visible to an on-path adversary; when browsing is server-side (e.g., OpenAI's Operator), only connections to the provider's infrastructure are exposed. We collect traces using both open-source and proprietary WRAs.

GPT Researcher [22]. GPT Researcher is a research agent designed for multi-hop investigation that decomposes prompts into sub-questions, gathers evidence, and synthesizes reports. We run it in `report_type="deep"` mode, enabling multi-

turn exploration across sources. GPT Researcher employs different LLMs for sub-tasks: `FAST_LLM` for lightweight operations (e.g., GPT-4o-mini, DeepSeek-V3-0324), `SMART_LLM` for reasoning and report generation (e.g., GPT-4.1, Mistral-7B-Instruct v0.3), and `STRATEGIC_LLM` for higher-level planning (e.g., o4-mini, Mistral-Nemo-12B). While the default configuration uses GPT-4, we also evaluate local LLM combinations to assess leakage outside proprietary ecosystems.

Browser-Use [46]. Browser-Use is an open-source agent that automates browsing through a local browser. By default, it issues Google queries via `search_web`, which frequently triggers reCAPTCHA; we therefore configure it to use Bing with automatic fallback. We further modify its system prompt to require at least five distinct page visits before summarization, preventing it from shortcutting tasks with prior knowledge or superficial searches. After each run, we extract domains and timestamps from the browser's recorded history to construct ordered traces. This setup makes Browser-Use suitable for evaluating how web agents' lightweight browsing patterns leak information.

AutoGen [89]. We configure AutoGen as a two-agent system consisting of an `AssistantAgent` and a `MultimodalWebSurfer`. Compared to Browser-Use, AutoGen is faster and more controllable, making it useful for both tasks. By default, AutoGen tends to summarize results directly without clicking through, so we enforce the same rule, visiting at least five pages. To collect metadata, we extend the web surfer into a custom `LoggingWebSurfer` that hooks Playwright APIs to record domains, timestamps, and IPs for all network events. Unlike GPT Researcher or Browser-Use, AutoGen logs not only primary domains but also auxiliary requests (ads, trackers, analytics), producing noisier but realistic traces that an actual network observer would see. We run AutoGen with GPT-4o for prompt recovery and Gemini 2.0 Flash for trait inference.

OpenAI Deep Research API. As a proprietary baseline, we evaluate the OpenAI Deep Research API. Unlike the interactive ChatGPT Deep Research interface, the API is stateless and relies entirely on instruction-rich inputs. To align its behavior with the UI, we provide **-DR** prompt variants (e.g., SESSION14-DR), rewritten to simulate the iterative refinement that would have been normally triggered by user interaction. Because browsing occurs remotely, only provider-level connections are visible locally; we nonetheless analyze these traces to benchmark inference difficulty against a high-capability proprietary system.

Rationale for Agent and Model Selection. We evaluate both research-oriented (GPT Researcher) and web-oriented (Browser-Use, AutoGen) agents to capture variation in browsing behavior and leakage. Each open-source agent is paired with both proprietary (GPT-4o, Gemini 2.0 Flash) and open-source (DeepSeek-V3, Mistral family) LLMs, reflecting realistic local deployments where users may expect stronger privacy guarantees. The Deep Research API serves as a commercial baseline. We deliberately exclude larger open-source models (e.g., LLaMA 3.1 70B, DeepSeek-R1), as they require substantial computational resources and often bypass web search by relying on internal knowledge, making them less suitable for our browsing-driven evaluation. Together, these choices ensure coverage of attacker-visible metadata across diverse agents and realistic local deployment settings.

Table 13: **Prompt recovery: 5-shot vs. contrastive ICL.**

Type of ICL	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
5-shot ICL	0.492	**0.415**	**0.770**	0.735	0.520	0.640
Simple (1 neg)	**0.495**	0.405	0.740	0.745	**0.525**	**0.645**
Simple (3 neg)	0.490	0.355	0.755	**0.750**	0.510	0.625
QF (1 neg)	0.493	0.385	0.765	0.680	0.520	0.590
QF (3 neg)	0.492	0.355	0.750	0.740	0.505	0.640

Table 14: **Prompt recovery: example selection strategies.**

Ex Selection Strategy	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
Random	**0.492**	**0.415**	**0.770**	0.735	**0.520**	**0.640**
Embedding-based	0.481	0.375	0.740	**0.755**	0.515	0.605

C.3 Inference Configuration

We evaluate our inference attacks using LLMs as an inference engine for few-shot ICL. A summary of ICL configurations is provided in Table 12.

Prompt Recovery. Each ICL prompt consists of five (trace, prompt) pairs, where the trace includes an ordered domain sequence and associated timing information, followed by an unseen trace for prediction. The format is fully natural language, with traces presented as newline-separated domain lists (and timing when present). Examples are drawn from distinct topics to prevent overfitting and to test generalization across different intent types. This format encourages the model to infer high-level task semantics from the structure, content, and temporal progression of visited domains.

Trait Inference. For trait inference, we adopt a 3-shot ICL format. Each example provides a domain trace and corresponding persona traits. The reduced shot count reflects the greater regularity of trait-disclosing behavior over time and helps minimize prompt length for complex structured outputs. The model is asked to generate a structured trait list in the format `- Trait: Value`, and is explicitly instructed to infer as many traits as possible, even under partial evidence.

D Additional Prompt Recovery Results

Effect of ICL Configurations. Contrastive ICL [23] showed that adding negative examples to the in-context setup can im-

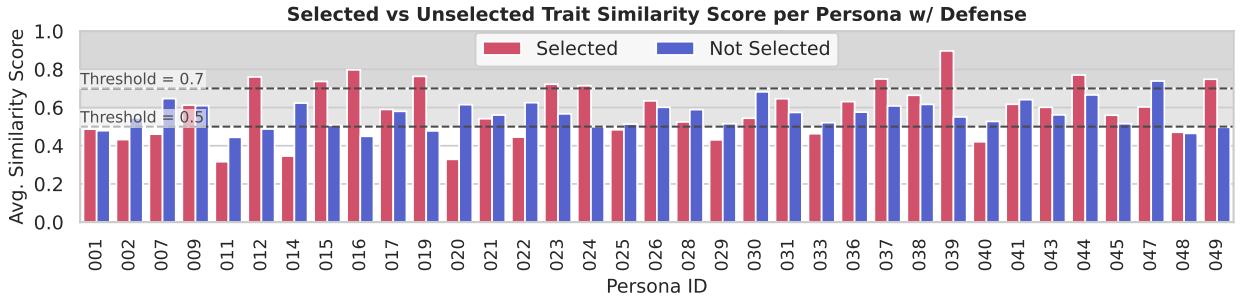

Figure 8: Average inference accuracy for selected vs. unselected traits per persona, showing generally lower accuracy than Figure 5 with the proposed virtual persona defense applied.

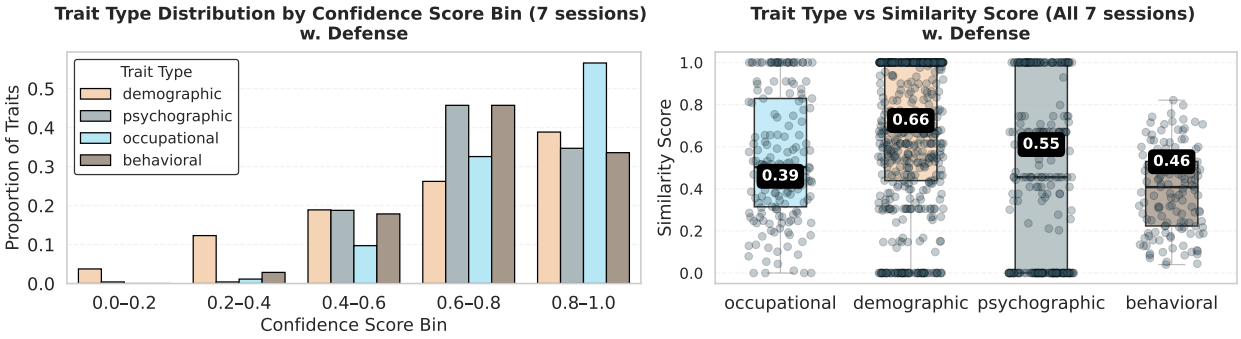

Figure 9: Trait-level breakdown of inference performance with the virtual persona defense. Confidence distributions remain similar to Figure 6, but average similarity scores are lowered across all categories.

Table 15: **Prompt recovery: different example orderings**

Ex Ordering	SBERT	LLM-Judge	OBELS			
			E_{func}	E_{dom}	E_{sem}	T_{ent}
Random	0.481	0.375	0.740	**0.755**	0.515	0.605
Ascending	**0.493**	**0.415**	0.725	0.735	**0.525**	**0.645**
Descending	0.481	0.380	**0.755**	0.745	0.495	0.620

prove performance, even when negatives must be synthesized by the LLM. In our baseline, each 5-shot prompt includes five (trace, prompt) pairs followed by a new trace for prediction. Contrastive variants augment these with one or three negatives, either in a simple setup or in our proposed **Quality-Filtered (QF) contrastive ICL**, where negatives are regenerated until their SBERT similarity to the original prompt falls below a threshold, ensuring sufficient dissimilarity.

As shown in Table 13, adding negatives did not improve overall results. While domain-type alignment (E_{dom}) occasionally improved, most other metrics decreased. Even with larger numbers of negatives and QF filtering, performance remained below the 5-shot baseline. We attribute this to the difficulty of the prompt inference task: the model may already operate near its limits, so additional negatives increase confusion rather than sharpening topical discrimination.

Effect of Example Selection. We compare random versus embedding-based selection of ICL examples. In the random strategy, examples are sampled from the training set for each test instance. For embedding-based selection, all training and test traces are encoded with the `all-MiniLM-L6-v2`[1] sentence transformer, and the training examples with the highest cosine similarity to the test trace are chosen. As shown in Table 14, embedding-based selection does not improve prompt recovery accuracy and, in some metrics, performs slightly worse, suggesting that random sampling, by providing more diverse examples, is the more effective strategy.

Effect of Example Ordering. We test whether the ordering of embedding-selected examples influences ICL performance, comparing random order, ascending similarity, and descending similarity. As shown in Table 15, ordering has little effect: scores remain close across all metrics. This suggests that once relevant examples are chosen, their sequence contributes minimally to recovery quality.

E Additional Virtual Persona Defense Results

Figure 8 and Figure 9 provide additional experimental results of the proposed virtual persona defense beyond the median similarity score reductions reported in the main text.

[1]Fine-tuned on 1B sentence pairs with a contrastive objective, this model maps each trace into a 384-dimensional vector space.