Scalable PII Discovery in Mobile App Databases via Sample-Guided Search

Your N. Here
Your Institution

Second Name
Second Institution

Abstract

Discovering personally identifiable information (PII) in mobile forensic databases is a core investigative task that supports rapid triage, identity attribution, and cross-application linkage, yet remains difficult because app data is spread across many SQLite schemas, stored in poorly documented layouts, and often buried inside free text or semi-structured fields. We cast PII discovery as a hypothesis-driven search problem under uncertainty and introduce an agent-guided workflow that alternates between low-cost exploration and targeted extraction. The system repeatedly proposes candidate table-column locations, issues sample-based probes to gather small observations, and then escalates to full extraction only when confidence is sufficient. Planning decisions are guided by schema metadata plus execution feedback, allowing the framework to prioritize semantically plausible regions without assuming exhaustive coverage. We evaluate this approach on heterogeneous Android and iOS datasets drawn from real-world mobile applications. The system recovers multiple PII categories including email addresses, phone numbers, usernames, person names, and postal addresses, even when the evidence is embedded inside unstructured content. At the same time, the sample-guided strategy prunes the column-level extraction workload by more than 90 percent across applications, supporting scalable discovery under bounded investigative budgets. Overall, uncertainty-aware, hypothesis-driven search offers a practical complement to deterministic extraction pipelines for modern mobile forensics.

1 Introduction

Mobile devices are central evidence sources in security investigations. They store large volumes of application data in heterogeneous, app-specific databases—most commonly SQLite—with frequent schema churn and limited public documentation [3,5]. Within these databases, personally identifiable information (PII) such as email addresses, phone numbers, usernames, person names, and postal addresses often provides investigation-critical pivots for cross-app linkage, attribution, and scoping potential exposure. Such PII also anchors security workflows beyond forensics, including incident response and fraud or account-takeover investigations. Yet, reliably locating and extracting PII remains difficult because evidence is scattered across semi-structured and free-text fields and represented in non-standard ways [15].

A key obstacle is that evidence rarely resides in clean, dedicated fields. Applications embed PII in message bodies, serialized objects, logs, caches, and auxiliary tables whose semantics may be unclear or misleading [10]. Even pattern-like PII (e.g., emails or phone numbers) may be fragmented, inconsistently normalized, or encoded in ways that degrade regex-only scanning; names and postal addresses are often harder still because they lack stable surface patterns. As a result, analysts either manually inspect large portions of a database or apply broad scanning that produces substantial noise. Exhaustive scanning approaches, including stream-based triage tools [7], are theoretically complete but increasingly impractical for large app corpora and low-density evidence under typical investigative time constraints.

These conditions make PII discovery a search problem under uncertainty: an analyst often does not know whether a target PII type exists, where it is stored, or how it is represented. This motivates intelligent orchestration [17] and the use of Large Language Models (LLMs) to help interpret heterogeneous artifacts [11,16], while still bounding computation and analyst effort.

A practical requirement in forensic and incident-response settings is auditability: analysts must be able to justify what data was examined and how each reported entity was obtained. We therefore constrain all database access to deterministic, read-only query tools and record provenance that links each extracted entity to its database, table, column, and retrieval mode. This design supports reproducible triage while keeping LLM reasoning grounded in observable samples rather than uncontrolled database access. This is especially important when results may be used for attribution, legal process, or high-stakes response decisions.

We address this challenge by formulating PII discovery as a hypothesis-driven search process with two stages: *PII exploration* and *PII extraction*. Exploration proposes and tests hypotheses about likely PII-bearing regions using sample-aware probing. Extraction performs targeted retrieval and normalization only after a region is validated with sufficient confidence. This workflow is intended to *complement* targeted extraction: when schemas and artifact locations are known, application-specific parsers remain preferable, while our method is most useful as an initial discovery step that narrows the search space and surfaces plausible sources for follow-on validation and precise extraction.

To operationalize this perspective, we introduce an agent-guided architecture in which a planner orchestrates LLM-assisted exploration and extraction under explicit control flow and deterministic query execution, enabling confidence- based escalation and principled stopping under bounded budgets.

We evaluate the approach across multiple PII types—email addresses, phone numbers, usernames, person names, and postal addresses—on Android and iOS application datasets.

This work makes the following contributions:

- We formulate scalable PII discovery in mobile application databases as a hypothesis-driven search problem under uncertainty, explicitly distinguishing between PII exploration and PII extraction.

- We propose an agent-guided architecture that operationalizes this formulation through LLM-assisted planning, sample-aware exploration, and controlled extraction with confidence-based escalation.

- We empirically evaluate the proposed approach on heterogeneous mobile forensic databases across Android and iOS applications, reporting both discovery effectiveness and reduction of the effective extraction search space across multiple PII types.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 formulates the search problem and system overview. Section 4 describes the workflow. Section 5 presents evaluation and threats to validity. Section 6 discusses limitations and implications, and Section 7 concludes.

2 Related Work

The discovery of personally identifiable information (PII) in mobile ecosystems has evolved from manual data parsing to increasingly automated and reasoning-driven exploration. This section situates our work within prior research on mobile forensic databases, traditional PII extraction techniques, and emerging agentic approaches to digital forensics. We emphasize a persistent gap that motivates our hypothesis-driven formulation: in many investigations, analysts must first *localize* likely PII-bearing regions under uncertainty before any targeted, app-specific extraction can be applied.

2.1 Mobile Forensic Databases

Mobile forensic investigations frequently rely on application-specific SQLite databases to recover artifacts such as messages, contacts, and logs. Prior work has focused on systematically extracting and interpreting these databases, often through app-specific reverse engineering or automated database identification. Daraghmi *et al.* [6] propose FORC, an automated framework for identifying SQLite databases on Android devices, including those lacking standard file extensions. Other studies examine the forensic artifacts of individual applications, such as messaging platforms, by analyzing database schemas and storage behaviors [3].

While these approaches improve coverage and automation of database extraction, they often assume that relevant evidence can be identified via schema inspection or predefined artifact locations. In practice, mobile application databases contain heterogeneous schemas and free-text fields in which PII is sparse, informally represented, or embedded in semantically ambiguous tables. This semantic gap motivates discovery strategies that reason beyond structural metadata alone and that can prioritize where to search before performing targeted extraction.

2.2 PII in Digital Forensics

The identification of PII is a recurring task in digital forensic investigations, supporting objectives such as user attribution, activity reconstruction, and privacy violation analysis. PII in mobile databases spans both structured and semi-structured forms, including email addresses, phone numbers, usernames, person names, and postal addresses.

Traditional approaches rely on pattern-based extraction techniques, such as `bulk_extractor` [7], which perform high-throughput, stream-based scanning to extract features directly from raw media. While effective for early-stage triage, exhaustive techniques can generate substantial noise and provide limited contextual understanding of evidentiary relevance, particularly when representations are heterogeneous or encoded.

More recent work has explored the use of large language models for semantic PII extraction. Liu *et al.* [11] evaluate LLM-based approaches for personal information extraction, demonstrating strong performance on text-centric inputs while highlighting the need for verification mechanisms to mitigate hallucinations. However, these approaches typically assume that relevant text regions have already been localized, whereas our work focuses on the preceding problem of finding likely PII-bearing regions within large, heterogeneous mobile databases.

Related work has also examined structured forensic evidence acquisition and preservation once relevant artifacts are known. Xu *et al.* propose a forensic evidence acquisition model for data leakage attacks that emphasizes integrity and controlled collection [21]. In contrast, our work targets discovery under uncertainty and is intended to complement, not replace, targeted parsers and app-specific extraction methods when artifact locations are already established.

2.3 Automated Evidence Discovery and Agentic Reasoning

Several systems have explored automation to assist forensic evidence discovery. Static analysis techniques, such as EviHunter [5] and related approaches [10], infer evidentiary storage locations by analyzing application code. While precise, such methods may be brittle under application updates, obfuscation, or closed-source components, and they do not directly address search-space control when evidence is embedded in free-text or semi-structured fields.

More recent research has shifted toward intelligent, agent-based forensic systems. Wickramasekara and Scanlon [17] investigate the use of AI agents to orchestrate complex forensic workflows. Mukherjee and Kantarcioglu [12] propose PROVSEEK, an agentic framework that leverages LLMs for provenance-driven investigations, emphasizing hypothesis refinement and evidence validation. Sharma *et al.* [16] further demonstrate the effectiveness of locally deployed LLMs optimized for forensic reasoning.

Complementary work has explored post-discovery reasoning and evidence representation. Xu and Xu propose knowledge-graph-based techniques for visualizing and reasoning about presentable digital forensic evidence after extraction [20], while Addai *et al.* investigate graph-based approaches for analyzing relationships among discovered evidence artifacts across devices [1]. These approaches operate downstream in the forensic pipeline and generally assume that relevant evidence has already been identified.

Recent studies further demonstrate the potential of LLMs to support forensic automation and semantic reasoning. Xu *et al.* examine how LLMs can transform digital forensic workflows through improved interpretation and automation [19], while Zhou *et al.* employ LLMs to construct forensic evidence networks for cybercrime insight generation [22]. Concurrent work has highlighted reliability and hallucination risks of LLMs in forensic contexts [9, 18], underscoring the importance of constrained reasoning and auditable execution.

In contrast to prior work that emphasizes acquisition, interpretation, behavioral reconstruction, or post-hoc relationship analysis, this paper frames PII evidence discovery as a hypothesis-driven search problem under uncertainty. Our approach explicitly targets the challenge of prioritizing and validating likely evidence-bearing regions within large, heterogeneous mobile forensic databases, bridging the gap between high-throughput triage techniques and downstream forensic reasoning systems.

3 Problem Formulation and Design Goals

This work studies *PII discovery* in heterogeneous mobile application databases encountered in forensic investigations. Unlike conventional extraction tasks where target artifacts and fields are known in advance, PII discovery operates under substantial uncertainty. The analyst often does not know (i) whether a target PII type exists in the dataset, (ii) where it is stored across tables and columns, or (iii) how it is represented within semi-structured or encoded content. Treating PII discovery as a single deterministic scan or a fixed set of rules is therefore brittle and inefficient at scale.

We formulate PII discovery as a hypothesis-driven search process under uncertainty. The system iteratively proposes testable hypotheses about likely PII-bearing database regions, probes them using small samples, updates its belief state based on observed evidence, and escalates to full extraction only when the evidence likelihood is sufficiently high. This approach complements targeted extraction by serving as an initial discovery step when schemas and artifact locations are unknown.

3.1 PII Discovery as Search Under Uncertainty

Let D denote a mobile forensic database (typically SQLite). Let E be a target PII type. We consider common forensic PII types such as *email address*, *phone number*, *username*, *person name*, and *postal address*. A *region* is a table–column pair $r = (T, C)$, optionally annotated with a schema-provided type hint $\tau(C)$ (e.g., TEXT/INTEGER/BLOB) when available, which we treat as a weak cue rather than a reliable semantic label.

PII discovery is challenging because mobile applications rarely store PII in dedicated, clearly labeled fields. PII may appear in free-text message bodies, serialized JSON, application logs, or binary-like blobs and escaped strings. It may also be stored in informal, partial, or encoded forms that defeat simple pattern matching. This yields three practical uncertainty dimensions:

- **Existence uncertainty:** the database may not contain the target PII type at all.
- **Location uncertainty:** if present, PII may be scattered across many tables and columns, including auxiliary and misleading fields.
- **Representation uncertainty:** values may be canonical, informal, mixed with other content, or encoded/escaped in ways that obscure patterns.

Table 1 provides representative examples of heterogeneous PII representations that motivate a search-based formulation. Even when an entity type is conceptually pattern-like (emails or phone numbers), the stored representation may include escaping or surrounding noise that makes regex-only scanning incomplete or overly noisy. For person names and postal addresses, which are inherently variable and context-dependent, pattern matching alone is often error-prone.

Two-stage view. Exhaustive scanning across all regions is increasingly impractical as database size grows and the density of relevant evidence decreases. We therefore separate PII discovery into two stages:

- **PII exploration:** identify and validate candidate PII-bearing regions using sample-aware probing, producing evidence-likelihood judgments with confidence.
- **PII extraction:** once a region is validated, perform targeted, comprehensive retrieval and normalization for the confirmed source region(s).

Hypotheses. A *hypothesis* is an explicit, testable claim that a region $r = (T, C)$ is likely to contain instances of PII type E. Hypotheses are generated during exploration, evaluated using samples from r, and then refined, discarded, or escalated to extraction based on confidence.

Definition (PII Discovery). Given a database D, a target PII type E, and an exploration budget B, PII discovery seeks a set of validated regions $R^\star$ that are likely to contain E by adaptively selecting regions to probe (subject to B) and escalating only high-confidence regions to extraction.

3.2 Design Goals

To operationalize PII discovery as search under uncertainty, we adopt the following design goals:

- **Adaptive hypothesis generation:** iteratively propose and refine candidate regions based on observed signals and prior outcomes.
- **Sample-aware exploration:** probe regions using limited samples to estimate evidence likelihood before performing full extraction.
- **Robustness to heterogeneous representations:** accommodate informal and mixed-content fields, serialized structures, and encoded or escaped values.
- **Explicit uncertainty management:** guide escalation and termination using confidence and budget-aware stopping conditions.
- **Auditability and reproducibility:** ensure all database access is performed by deterministic tools, and reasoning decisions are traceable to observable probe results.

These goals motivate the planner-driven, sample-aware architecture described next.

4 Methodology

This section describes how the proposed system operationalizes hypothesis-driven PII discovery under uncertainty. As formulated in §3, the system separates *PII exploration* (sample-aware probing and validation) from *PII extraction* (targeted, comprehensive collection and normalization), with a planner controlling escalation under a bounded exploration budget. We focus on the operational details of planning, sample-aware probing, and confidence-based escalation.

4.1 Workflow Overview and Control Flow

We implement discovery as a state-machine-guided workflow organized as a DAG (Figure 1). A planner proposes hypotheses over candidate table–column regions, requests sample-aware probes via deterministic query tools, and updates its context from exploration feedback. High-confidence regions (validated when `confidence` $\geq \tau$) are escalated to extraction for full retrieval, entity recovery, normalization, and deduplication. The workflow terminates when extraction completes for at least one validated region or when the exploration budget is exhausted. The same workflow is applied independently per PII type E, producing type-specific validated regions and extracted entity sets with shared provenance structure.

4.2 Planner and Probing

The planner generates and ranks hypotheses over regions $r = (T, C)$ using deterministic schema metadata (table/column names and optional type hints), combining lexical cues (for example, `email`, `phone`, `addr`, `user`), structural cues (message/log/profile tables and text-like columns), and cross-table cues (identifier-like connectors). It maintains a context $\mathcal{K}$ of tested regions and outcomes to avoid redundant probing, and enforces an exploration budget B that bounds the number of probed regions and sampled rows per region.

All database access is read-only and mediated by deterministic query tools. For each candidate region, the system retrieves a small sample $\mathcal{S}_r$ via row-limited or distinct-value probes with lightweight filtering for null/empty values. If a region is validated, the system retrieves the complete value set $\mathcal{V}_r$ for extraction. The LLM does not execute SQL; it operates only on values returned by deterministic queries.

4.3 PII Exploration

Given $(E, r, \mathcal{S}_r)$, exploration returns a structured record

$$\mathrm{Explore}(E, r, \mathcal{S}_r) \to (\texttt{evidence\_likely}, \texttt{confidence}, \texttt{rationale}, \texttt{excerpts}). \quad (1)$$

Table 1: Representative examples of heterogeneous PII representations in mobile app databases that motivate hypothesis-driven discovery.

ID	Probe sample from region $r = (T, C)$ (shown as `C: value`)	Target PII type(s) E
1	`payload: "...\n\x19`**`heisenbergercarro@gmail.com`**`\x12..."`	email address
2	`content: "...\n\x0c`**`+16506808040`**`\x12\t\n..."`	phone number
3	`full_name:` **`"Marsha Mellos"`**	person name
4	`user: "mmellos"`	username
5	`data:` **`"Meet at 1500 Market St SF, call 4435189592"`**	postal address, phone number

where $0 \leq$ `confidence` ≤ 1 is a scalar score. LLM assistance is used to handle representation uncertainty in semi-structured or noisy values (for example, names and postal addresses in free text, or escaped/encoded emails and phone numbers). Escalation is conservative: the planner authorizes extraction only when confidence exceeds a threshold τ (optionally confirmed by a second probe).

4.4 PII Extraction and Outputs

For each validated region r, extraction identifies instances of type E in $\mathcal{V}_r$ and outputs normalized entities with provenance:

$$\text{Extract}(E, r, \mathcal{V}_r) \rightarrow \{(e, \text{meta}(e))\}.$$

Normalization is conservative (for example, email casing/whitespace, digit normalization for phone numbers, and whitespace cleanup for names and postal addresses) and preserves original evidence strings for traceability. Deduplication merges identical canonical entities while aggregating provenance. Provenance records at least database identifier (if applicable), table T, column C, and a descriptor distinguishing exploration sampling from extraction retrieval.

5 Evaluation

This section evaluates the proposed hypothesis-driven PII discovery framework on mobile forensic datasets. We conduct evaluation at the *application level*, aggregating results across the selected databases for each application. We assess effectiveness, efficiency, robustness, and model sensitivity through four research questions, each addressed in a dedicated subsection.

5.1 Empirical Study Design

Devices and Source Dataset. The SQLite databases analyzed in this study were sourced from the Cellebrite 2024 Capture-the-Flag (CTF) dataset [14]. We use two devices from the dataset: a Samsung Galaxy S21 running Android 14 and an iPhone 11 Pro running iOS 17.5.1. Both devices have 256 GB of internal storage and were acquired using Full File System (FFS) extraction, yielding forensic images that include user-installed applications, system services, and associated application data. The devices belong to two simulated suspects, referred to as *Otto* (Android) and *Sharon* (iOS).

Table 2 summarizes the overall scale of the extracted images. From this full corpus, we select 10 representative applications (five per platform) and up to three databases per application for controlled evaluation (25 databases total).

Table 2: Overall dataset scale across extracted mobile applications and databases.

Device	Applications	Databases	Tables
Android (Galaxy S21)	500	1531	15,721
iOS (iPhone 11 Pro)	158	1504	14,753

Application Selection. We select commonly encountered application categories on each platform, including messaging, social networking, web browsing, location services, and system utilities. Applications are identified using platform-specific package identifiers. On Android, we evaluate WhatsApp (`com.whatsapp`), Snapchat (`com.snapchat.android`), Telegram (`org.telegram.messenger`), Google Maps (`com.google.android.apps.maps`), and Samsung Internet (`com.sec.android.app.sbrowser`). On iOS, we evaluate WhatsApp (`net.whatsapp.WhatsApp`), Contacts (`com.apple.AddressBook`), Apple Messages (`com.apple.MobileSMS`), Safari (`com.apple.mobilesafari`), and Calendar (`com.apple.mobilecal`). For readability, tables report only application names rather than package identifiers.

Database Selection. Within each selected application, we analyze a bounded subset of SQLite databases. We rank candidate databases using (1) file size and (2) recency (file modification timestamps in the extracted image), and break ties using (3) semantic cues in filenames (e.g., `message`, `chat`, `store`, `profile`) and (4) forensic domain knowledge of typical application data organization. For each application, we

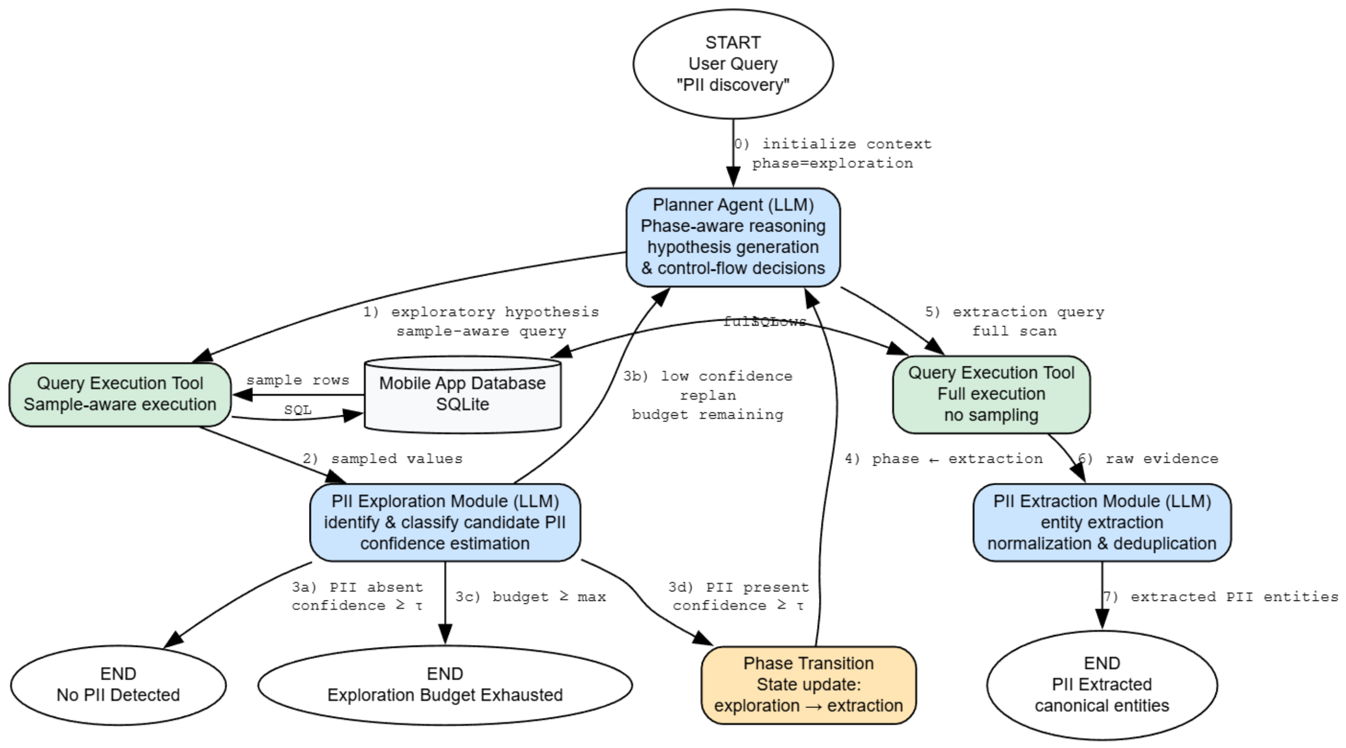

Figure 1: Agent-guided control flow for hypothesis-driven PII discovery. A planner agent controls hypothesis generation and state transitions. Sample-aware execution supports PII exploration, and full execution enables PII extraction. Deterministic tools mediate all database access, and explicit transitions govern termination.

select up to the top three databases under this ranking. Selection is performed prior to any content-level analysis and does not depend on whether PII is ultimately discovered, avoiding outcome-driven bias.

Table 3 lists the analyzed applications and selected databases (25 total across ten applications). Each application is assigned a unique identifier (ID) used consistently across subsequent evaluation tables.

Distinct PII Entities. We report counts of *distinct* PII entities computed over a lightweight canonical form. For each extracted instance x, we compute $\mathsf{canon}(x)$ using type-agnostic cleanup (trim whitespace) and type-specific canonicalization: email addresses are lowercased; phone numbers remove non-digit characters and preserve a leading + when present; and person names are trimmed and case-folded. Two instances are treated as the same entity (counted once) if their canonical forms are identical.

Distinctness is computed within a specified scope. We report (i) **database-level** distinct counts (within one database file), (ii) **application-level** distinct counts (union across selected databases per application, then deduplicate), and (iii) **corpus-level** distinct counts (union across all evaluated applications and databases, then deduplicate). Unless stated otherwise, results are reported at the application level.

Ground Truth. Ground truth is established through manual inspection of the selected application databases and cross-checked against known Cellebrite CTF challenge artifacts when available. For each PII type, we curate the set of distinct entities present in these databases, including entities embedded in free-text fields and entities recovered from deserialized application data structures (e.g., recoverable BLOB-derived payloads). Annotation is performed by three annotators (one industry practitioner and two students with computer science and digital forensics training). Each database is independently reviewed by at least two annotators; disagreements are resolved through joint adjudication and a final consensus pass. Encrypted or irrecoverable payloads and deleted artifacts are excluded from ground truth. When reporting distinct entity counts, ground truth entities are canonicalized using the same $\mathsf{canon}(\cdot)$ procedure described above. The established ground truth for the databases listed in Table 3 is provided in Appendix 7 (Table 13).

LLM Configuration. We instantiate the planner, exploration, and extraction components with the GPT-4o-mini model, using temperature 0 and fixed prompts and tool interfaces across runs to support reproducibility. We report the number of runs per setting and any model-sensitivity analyses in the corresponding research question subsections.

Table 3: Applications and selected databases for PII discovery (25 total). We report base filenames; extensions vary by platform.

ID	Device	Application	App Type	Selected Databases
A1	Android	WhatsApp	Communication	`wa`, `msgstore`, `commerce`
A2	Android	Snapchat	Ephemeral Messaging	`main`, `core`, `journal`
A3	Android	Telegram	Cloud Messaging	`cache4` (three instances for three accounts)
A4	Android	Google Maps	Location Services	`gmm_storage`, `gmm_myplaces`, `peopleCache_ACCOUNT`
A5	Android	Samsung Internet	Web Browser	`SBrowser`, `SBrowser2`, `searchengine`
I1	iOS	WhatsApp	Communication	`ChatStorage`, `ContactsV2`, `CallHistory`
I2	iOS	Contacts	System Contacts	`AddressBook`, `AddressBookImages`
I3	iOS	Apple Messages	Communication	`sms`
I4	iOS	Safari	Web Browser	`History`, `CloudTabs`
I5	iOS	Calendar	Scheduling	`Calendar`, `Extras`

5.2 RQ1: Effectiveness of PII Discovery Across Mobile Applications

RQ1: How effectively does the proposed system perform PII discovery (i.e., exploration plus extraction) across mobile applications on Android and iOS devices?

This question evaluates the proposed hypothesis-driven approach across heterogeneous mobile applications without relying on exhaustive database scanning. Effectiveness is assessed along two complementary dimensions: (i) the volume and diversity of PII recovered across applications, and (ii) the forensic plausibility of the discovered evidence within realistic application contexts.

Table 4 reports the number of *distinct* PII entities extracted per application and PII category across Android and iOS platforms, using *application-level* distinctness as defined in Section 5.1.

Overall, the results show that the system consistently uncovers multiple PII types across diverse applications and platforms, including cases where evidence is embedded in unstructured or semi-structured fields rather than cleanly typed schema attributes. For instance, in WhatsApp on iOS (I1), the system extracts a person name from conversational text stored in the `ztext` column (e.g., identifying "Rick" in the message "If you need anything give my man Rick a call"), illustrating its ability to recover PII from natural-language content. Recovered PII is highly concentrated in communication and identity-centric applications: Snapchat (A2), iOS WhatsApp (I1), and Contacts (I2) yield large volumes of identifiers, including over a thousand phone numbers and person names, reflecting the density of user-centric artifacts in messaging and contact ecosystems. In particular, the Contacts app (I2) contains over 1,000 saved contacts, and the system retrieves the phone numbers associated with these entries.

In contrast, several non-communication applications still yield non-trivial identity traces. Google Maps (A4) and Samsung Internet (A5) reveal user-linked identifiers through cached metadata and history artifacts, indicating that PII exposure extends beyond explicit contact or messaging stores. Finally, Telegram (A3) yields zero recoverable PII under our workflow because relevant fields in the sampled tables are stored as encrypted payloads (e.g., BLOBs) with insufficient plaintext context for reliable extraction.

These counts reflect the outcome of the full PII discovery process: candidate PII-bearing regions are first identified during PII exploration and then resolved into concrete, validated PII entities during PII extraction. Notably, several messaging and social applications whose database schemas provide limited semantic guidance to investigators (e.g., WhatsApp, Snapchat, and Apple Messages) still produce substantial PII yields [4]. Prior work has shown that, in such applications, relevant evidence is often embedded within free-text fields, auxiliary tables, or columns whose semantic meaning cannot be inferred reliably from schema metadata alone [5]. These observations suggest that hypothesis-driven probing and sample-aware validation can surface PII that would likely be missed by purely schema-driven or keyword-only extraction pipelines.

To complement these quantitative findings, Table 5 presents representative examples of discovered PII artifacts for each application. The examples show that recovered PII is grounded in realistic application storage artifacts (e.g., message stores, address books, cached metadata, and browsing or calendar databases) rather than appearing as isolated tokens without context. The Telegram example highlights a common limitation: relevant fields may be stored as encrypted payloads (e.g., BLOBs), yielding no recoverable plaintext PII.

5.3 RQ2: Extraction Search Space Reduction via Hypothesis-Driven Planning

RQ2: To what extent does hypothesis-driven planning during exploration reduce the *extraction search space* (the set of columns exhaustively scanned during row-level PII extraction), relative to all candidate columns in the selected databases for each application?

Table 4: PII Discovered Per Application and Database (ChatGPT 4o-mini)

ID	Application	Database	Email	Phone	User Name	Person Name	Postal Address	Total PII
A1	WhatsApp	commerce.db	0	0	0	0	0	**0**
		msgstore.db	0	7	5	15	0	**27**
		wa.db	0	16	680	10	0	**706**
A2	Snapchat	core.db	0	1	5	1	0	**7**
		journal.db	0	0	0	0	0	**0**
		main.db	1	13	0	11	0	**25**
A3	Telegram	account1cache4.db	0	0	0	0	0	**0**
		account2cache4.db	0	0	0	0	0	**0**
		account3cache4.db	0	0	0	0	0	**0**
A4	Google Maps	gmm_myplaces.db	0	0	0	0	0	**0**
		gmm_storage.db	0	0	0	0	0	**0**
		peopleCache_sh....db	2	0	2	0	0	**4**
A5	Samsung Internet	SBrowser.db	1	0	5	0	0	**6**
		SBrowser2.db	0	0	0	0	0	**0**
		searchengine.db	0	0	17	0	0	**17**
I1	WhatsApp	CallHistory.sqlite	0	0	10	0	0	**10**
		ChatStorage.sqlite	0	0	0	14	2	**16**
		ContactsV2.sqlite	0	655	0	0	0	**655**
I2	Contacts	AddressBook.sqlitedb	6	548	1	748	0	**1303**
		AddressB....sqlitedb	0	0	0	0	0	**0**
I3	Apple Messages	sms.db	1	0	0	10	0	**11**
I4	Safari	CloudTabs.db	0	0	25	0	0	**25**
		History.db	0	0	4	0	0	**4**
I5	Calendar	Calendar.sqlitedb	1	0	0	0	0	**1**
		Extras.db	0	0	0	0	0	**0**

This research question evaluates the efficiency benefits of the proposed approach by quantifying how much exhaustive examination can be avoided during PII discovery. As discussed in the introduction, practical forensic constraints limit the extent to which large and heterogeneous datasets can be exhaustively examined, motivating adaptive strategies that balance discovery effectiveness with computational and investigative cost.

It is important to distinguish between the agent's *reasoning space* and the *extraction search space* measured in this evaluation. During the PII exploration phase, the planner agent reasons over the complete schema of the selected databases for an application (all tables and columns) and may inspect limited sample rows to inform hypothesis generation. In this phase, the full schema constitutes the agent's reasoning space and is not artificially restricted or pruned.

In contrast, the *effective extraction space* is the subset of database columns that are ultimately subjected to exhaustive row-level inspection during PII extraction. Although the agent reasons globally over the full schema during exploration, only columns selected as likely evidence-bearing candidates are included in the effective extraction space. This distinction allows search space reduction to be quantified as an execution-level efficiency metric without constraining the agent's reasoning or hypothesis generation capabilities.

Table 6 reports the resulting reduction in effective extraction space across the selected Android and iOS applications. For each application, we report the total number of candidate columns across all tables in the selected databases, the number of columns exhaustively scanned during row-level PII extraction, and the corresponding percentage reduction. Because extraction scans all rows for each selected column, the execution cost can also be expressed as the number of scanned cells, computed as $\sum_t |R_t| \cdot |C_t^{\text{sel}}|$, where $|R_t|$ is the row count of table t and $|C_t^{\text{sel}}|$ is the number of selected columns in table t. Across applications with diverse schema sizes and structures, hypothesis-driven planning consistently prunes a large fraction of columns from exhaustive examination, often exceeding a 90% reduction. Applications with large and heterogeneous schemas, such as messaging and browser applications, exhibit particularly strong reductions, demonstrating that the approach scales favorably with schema complexity. These results indicate that the proposed method substantially reduces the amount of data that must be exhaustively exam-

Table 5: Illustrative Examples of PII Artifacts Identified During PII Discovery

ID	Apps	Representative PII Discovered From DBs
A1	WhatsApp	msgstore.db (PERSON_NAME): Mary Garcia wa.db (PHONE): 8624338328
A2	Snapchat	main.db (USERNAME): abe_rudder main.db (PHONE): 2025692832 main.db (PERSON_NAME): Fendi Dinero
A3	Telegram	No recoverable PII observed in sampled tables; fields contained encrypted payloads (BLOBs).
A4	Google Maps	peopleCache_sharononeil368@gmail.com _com.google_14.db (PHONE): 17423794330
A5	Samsung Internet	SBrowser.db (EMAIL): sharononeil368@gmail.com SBrowser.db (USERNAME): sharononeil368
I1	WhatsApp	ChatStorage.sqlite (PHONE): 2027132090 ChatStorage.sqlite (PERSON_NAME): Fate
I2	Contacts	AddressBook.sqlitedb (EMAIL): edventure77@gmail.com AddressBook.sqlitedb (PHONE): 1003163800
I3	Apple Messages	sms.db (PERSON_NAME): Anya sms.db (PHONE): 2109299940
I4	Safari	History.db (USERNAME): hersheypark tickets
I5	Calendar	Calendar.sqlitedb (EMAIL): ottomatik1234@gmail.com

ined during PII discovery, while preserving the agent's ability to reason globally over the full database schema.

From a practitioner perspective, these findings show that hypothesis-driven planning can dramatically reduce the number of database fields that require row-level inspection, enabling investigators to focus their effort on a small, high-yield subset of application data under realistic time and resource constraints

5.4 RQ3: Robustness of Distinct PII Discovery Across Types and Applications

RQ3: How robust is the proposed approach across different PII types and data sources, in terms of (i) *corpus-level* distinct PII discovery and (ii) database-level coverage across the evaluated applications?

This research question evaluates whether the proposed hypothesis-driven approach generalizes across multiple categories of PII and diverse mobile applications. We focus on recovery of *distinct, investigator-relevant identifiers* rather than exhaustive enumeration of all occurrences, since investigations often prioritize unique leads over redundant repeats.

Metrics. We assess robustness using two complementary classes of metrics. First, we measure *corpus-level entity performance*. Let $\mathcal{G}$ be the set of distinct PII entities in ground truth across the full corpus, and let $\mathcal{S}$ be the corresponding set produced by the system under the same canonicalization rules. We report distinct recall $|\mathcal{G} \cap \mathcal{S}|/|\mathcal{G}|$ and distinct precision $|\mathcal{G} \cap \mathcal{S}|/|\mathcal{S}|$ (Table 7).

Table 6: Reduction of effective extraction space via hypothesis-driven planning. Total candidate columns are counted over all tables in the selected databases for each application, while reductions correspond to columns exhaustively scanned during row-level PII extraction.

ID	Apps	Candidate Cols (Total)	Cols Scanned (Extraction)	Reduc. (%)
A1	WhatsApp	1637	14	99.14%
A2	Snapchat	848	2	99.76%
A3	Telegram	1197	0	100.00%
A4	Google Maps	80	2	97.50%
A5	Samsung Internet	185	11	94.05%
I1	WhatsApp	328	6	98.17%
I2	Contacts	13	13	0.00%
I3	Apple Messages	186	0	100.00%
I4	Safari	74	7	90.54%
I5	Calendar	541	0	100.00%

Second, we measure *database-level coverage*, which captures how broadly the proposed system discovers PII across databases where it exists. This metric complements distinct recall by evaluating robustness to source diversity and helps determine whether discoveries are distributed across many databases rather than concentrated in a single high-yield database. For a given PII type t, let $D_G(t)$ denote the set of databases that contain at least one entity of type t under ground truth, and let $D_S(t)$ denote the set of databases from which the system correctly extracts at least one entity of the same type. We define database-level coverage as

$$\text{Coverage}_{\text{DB}}(t) = \frac{|D_G(t) \cap D_S(t)|}{|D_G(t)|}.$$

This metric quantifies, for each PII type, the fraction of ground-truth-bearing databases covered by the system (Table 8).

To analyze how *database-level coverage* distributes across applications, let a denote an application and $D(a)$ the set of databases associated with that application. We define $D_G(a,t) = D(a) \cap D_G(t)$ and $D_S(a,t) = D(a) \cap D_S(t)$, and compute application-level coverage as

$$\text{Coverage}_{\text{App}}(a,t) = \frac{|D_G(a,t) \cap D_S(a,t)|}{|D_G(a,t)|}.$$

This metric measures, for each application and PII type, the fraction of ground-truth-bearing databases that are correctly covered by the system, thereby showing how coverage distributes across applications rather than being confined to a single application (Table 9).

Together, database-level and application-level coverage characterize the breadth of PII discovery across sources, while distinct recall captures extraction completeness within those sources.

Finally, to contrast database-level coverage with extraction completeness, we report per-application distinct recall (Table 10). For application a and PII type t, let $\mathcal{G}_{a,t}$ be the set of canonicalized ground-truth entities and $\mathcal{S}_{a,t}$ the canonicalized system output set aggregated across the application's databases. Per-type distinct recall is

$$R_{a,t} = \frac{|\mathcal{G}_{a,t} \cap \mathcal{S}_{a,t}|}{|\mathcal{G}_{a,t}|}.$$

We also report **All PII** recall by taking unions across types:

$$R_{a,\text{all}} = \frac{|(\bigcup_t \mathcal{G}_{a,t}) \cap (\bigcup_t \mathcal{S}_{a,t})|}{|\bigcup_t \mathcal{G}_{a,t}|}.$$

Results. Table 7 reports corpus-level distinct PII performance by type using canonicalized entity sets. Phone numbers and person names achieve high distinct recall (95.6% and 85.6%) with solid precision (87.0% and 78.3%). Email addresses show moderate recall (77.8%) but high precision (87.5%). User names have the weakest recall (35.1%) despite relatively high precision (83.3%), indicating many missed unique usernames. Postal addresses are fully recovered (100.0% recall) but with very low precision (20.0%), reflecting substantial over-extraction relative to the small ground-truth set. The largest gap appears for user names, where low distinct recall suggests that many user identifiers are stored in heterogeneous formats or columns that are not consistently captured by the extraction rules. Postal addresses exhibit low precision because the ground-truth set is very small, making precision sensitive to a few extra address-like false positives.

Table 8 summarizes database-level source coverage for each PII type across the corpus. Overall, the system achieves high coverage, recovering PII from the majority of databases in which it appears under ground truth. Phone numbers and person names exhibit strong coverage (88.9% and 85.7%), indicating that these identifiers are consistently discovered across most relevant databases. Email addresses also show high coverage (83.3%), while user names exhibit slightly lower coverage (75.0%), suggesting greater heterogeneity in how user identifiers are stored across databases.

Postal addresses achieve full coverage (100.0%), but this result reflects the presence of addresses in only a single ground-truth-bearing database, making coverage sensitive to small absolute counts. Notably, for all PII types the number of databases with system discoveries exceeds the number of ground-truth-bearing databases, indicating that the system explores broadly while coverage remains anchored to correct source recovery.

Table 9 reports application-level source coverage, $\text{Coverage}_{\text{App}}(a,t)$, as **covered/GT**, where covered $= |D_G(a,t) \cap D_S(a,t)|$ and GT $= |D_G(a,t)|$ ("-" if GT= 0). The table shows how PII discovery distributes across databases within each application. Across most applications, the system covers the majority of ground-truth-bearing databases for common PII types such as phone numbers and person names, with many applications achieving full coverage (e.g., 1/1 or 2/2). This indicates that PII discovery is not limited to a single database per application, but instead extends across multiple relevant sources.

Coverage varies by PII type and application. User names exhibit partial coverage in some applications (e.g., 1/2 in Snapchat), consistent with heterogeneous storage formats for user identifiers. Applications with no ground-truth PII of a given type are marked as not applicable. The *All PII* column summarizes coverage across types and shows that, whenever an application contains at least one PII-bearing database under ground truth, the system typically covers all such databases, demonstrating broad source coverage at the application level rather than concentration in a small number of sources.

Table 10 reports per-application distinct recall by PII type, measuring how completely unique ground-truth entities are recovered within each application after aggregation across its databases. Recall varies substantially across applications and PII types, reflecting differences in data presence and storage characteristics.

Phone numbers consistently achieve high distinct recall in applications where they appear under ground truth, including WhatsApp (0.92), WhatsApp iOS (0.97), Contacts (0.98), and Google Maps (1.00), indicating that this PII type is reliably extracted when present. Person names also exhibit strong recall in several applications (e.g., 0.80 for WhatsApp and 0.85 for WhatsApp iOS), though recall is undefined in applications where names do not occur in ground truth.

User names show more variable performance, with moderate recall in some applications (e.g., 0.55 for WhatsApp and 0.30 for Snapchat) and undefined recall in others where no ground-truth user names exist. This pattern is consistent with heterogeneous representations of user identifiers and application-specific storage practices. Email recall similarly varies by application, ranging from perfect recovery in Calendar (1.00) to partial recovery in Google Maps (0.85), and is undefined in applications without ground-truth email addresses.

Applications such as Telegram and Safari contain no ground-truth PII of the evaluated types, and therefore exhibit undefined recall across all categories. The *All PII* column aggregates recall across PII types and shows that, when an application contains PII-bearing entities, the system often recovers a large fraction of distinct identifiers (e.g., 0.90 for WhatsApp iOS and 0.93 for Contacts), even when recall varies by individual type. Together with the source coverage results, these findings indicate that broad database coverage does not necessarily imply uniform recovery of all distinct entities, underscoring the complementary roles of coverage and distinct recall in evaluating robustness.

Table 7: Corpus-level distinct PII performance by type. Counts are *unique* entities after canonicalization: GT ($|\mathcal{G}|$), System ($|\mathcal{S}|$), and Overlap ($|\mathcal{G} \cap \mathcal{S}|$). We report distinct recall and distinct precision for each type.

PII Type	**GT**	**System**	**Overlap**	**Recall**	**Precision**
Email Address	18	16	14	77.8%	87.5%
Phone Number	2091	2300	2000	95.6%	87.0%
User Name	4275	1800	1500	35.1%	83.3%
Person Name	2102	2300	1800	85.6%	78.3%
Postal Address	2	10	2	100.0%	20.0%

Table 8: Database-level source coverage across the 25 selected databases. For each PII type, we report the number of databases containing that type under ground truth (GT), the number of databases from which the system extracts at least one correct entity of that type, their overlap, and the resulting coverage ratio.

PII Type	**DBs with PII (GT)**	**DBs with discoveries (System)**	**Overlap**	**Coverage**
Email Address	6	7	5	83.3%
Phone Number	9	11	8	88.9%
User Name	4	7	3	75.0%
Person Name	7	10	6	85.7%
Postal Address	1	4	1	100.0%

Practical impact. From a practitioner's perspective, these results indicate that the approach can surface investigator-relevant *distinct* PII across heterogeneous mobile applications without requiring application-specific extraction rules, while avoiding exhaustive enumeration of redundant occurrences.

5.5 RQ4: Sensitivity to Language Model Choice

RQ4: How does the choice of language model affect (i) PII discovery yield and (ii) search efficiency under the same hypothesis-driven workflow?

This research question examines whether the effectiveness and efficiency of the proposed hypothesis-driven PII discovery workflow depend strongly on the choice of underlying language model. Rather than comparing models in terms of raw latency, cost, or benchmark accuracy, we evaluate the stability of discovery outcomes and pruning behavior when the framework is instantiated with different LLM backends. For additional context, we also report results from a traditional PII scanning baseline (`bulk_extractor-v1.6`) [7], which does not use hypothesis generation or column-level pruning.

We evaluate three representative models: GPT-4o-mini [13], Gemini 2.5 [8], and Qwen2.5-72B-Instruct [2]. Each model instantiates all LLM-dependent components of the workflow, including the planner agent's hypothesis generation and prioritization and the LLM-assisted modules used in PII exploration and PII extraction. All experiments are conducted under identical conditions, including the same datasets, stopping criteria, prompts, and deterministic execution tools. No model-specific heuristics or tuning are introduced.

Discovery yield. To assess yield, Table 11 reports *corpus-level distinct* PII counts by type, computed over the evaluated corpus using the distinctness definition in Section 5.1. For context, we also report a standard benchmark (BM) score (MMLU accuracy) for each model as a coarse indicator of general capability; this benchmark is not used by the workflow and is reported only to aid interpretation. The values in this table are *illustrative placeholders* to be replaced with results from the corresponding runs.

The overall mix of discovered PII types is stable across models, with modest variation in distinct entity counts per category. Such differences are consistent with small changes in hypothesis ordering and confidence-driven escalation decisions that affect which candidate regions are extracted within a bounded exploration budget.

Search efficiency. Table 12 summarizes search efficiency across methods using the same column-level effective search space definition as RQ2. For each application, we compute the fraction of candidate columns examined within its selected databases, and report the average search space reduction across applications (i.e., one minus the examined fraction).

Values in this table are *illustrative placeholders* to be replaced with results from the corresponding runs.

Across model backends, hypothesis-driven pruning is preserved: all instantiations substantially reduce the number of columns subjected to row-level extraction, with only modest variation in average columns examined. This suggests that the control-flow and tool-mediated probing strategy dominate the pruning behavior, while model choice primarily affects the ordering and confidence of intermediate hypotheses.

Validity check. To ensure that model-to-model differences are not driven by false positives, we perform a lightweight manual verification by tracing sampled extracted entities back to their originating database records and confirming that they correspond to genuine PII in context. (Verification sample

Table 9: Application-level source coverage by PII type. For each application and PII type, we report **covered/GT**, where GT is the number of databases within the application that contain at least one correct instance of the PII type under ground truth, and covered is the number of those databases from which the system correctly extracts at least one entity of the same type. A dash ("-") indicates that the PII type does not appear in ground truth for that application.

ID	**Application**	**Email** (covered/GT)	**Phone** (covered/GT)	**User Name** (covered/GT)	**Person Name** (covered/GT)	**Postal Address** (covered/GT)	**All PII**
A1	WhatsApp	-	1/2	1/1	1/1	-	2/2
A2	Snapchat	1/1	2/2	1/2	1/2	-	2/2
A3	Telegram	-	-	-	-	-	-
A4	Google Maps	1/1	1/1	1/1	-	-	1/1
A5	Samsung Internet	1/1	-	-	-	-	1/1
I1	WhatsApp (iOS)	-	2/2	-	2/2	1/1	2/2
I2	Contacts	1/1	1/1	-	1/1	-	1/1
I3	Apple Messages	0/1	1/1	-	-	-	1/1
I4	Safari	-	-	-	-	-	-
I5	Calendar	1/1	-	-	1/1	-	1/1

sizes and outcomes will be reported alongside the finalized RQ4 results.)

Overall, RQ4 indicates that the hypothesis-driven workflow is not tightly coupled to a specific language model. Instead, its performance is driven by the control flow and sample-aware validation governing PII exploration and PII extraction, allowing the framework to accommodate different LLM backends without altering the underlying discovery logic.

5.6 Threats to Validity

This evaluation has several limitations. First, results are based on two devices from a CTF corpus and may not reflect the full diversity of real-world usage and application versions. Second, although we manually construct ground truth over the selected databases, PII labeling remains imperfect for informal or obfuscated representations, and our scope excludes encrypted payloads and deleted artifacts. Finally, the system prioritizes efficient discovery of investigator-relevant *unique* PII under bounded budgets rather than exhaustive enumeration, and therefore does not guarantee identification of all PII instances.

6 Discussion

This section interprets the evaluation results in the context of mobile forensic practice. We discuss the practical implications of hypothesis-driven exploration and confidence-based escalation for efficient PII discovery in heterogeneous application databases, and summarize key method-level limitations of the proposed approach.

6.1 Implications for Mobile Forensics

Our results reinforce a common reality in mobile forensics: investigator-relevant PII is often embedded in free text, serialized objects, and logs rather than in clearly labeled schema fields. In such settings, purely schema-driven extraction or fixed, application-specific rules can miss evidence or require substantial manual effort to adapt to evolving schemas and representations.

By framing PII discovery as search under uncertainty, the proposed approach more closely matches how analysts work in practice. The planner guides attention toward semantically plausible regions, sample-aware probing enables rapid assessment without immediately committing to full scans, and confidence-based escalation provides an explicit mechanism for deciding when broader extraction is warranted. This supports timely triage and early lead generation, particularly when investigators prioritize distinct, attribution-relevant *unique* PII entities over redundant occurrences.

Importantly, the approach is not intended to replace exhaustive extraction when completeness is required. Rather, it provides a bounded-budget discovery layer that can prioritize likely evidence-bearing regions, reduce analyst workload, and serve as a principled entry point for deeper analysis. Because probe results and escalation decisions are explicit and loggable, the approach also improves transparency into what was examined and why, which is valuable for reporting and tool accountability.

6.2 Method Limitations

Beyond the evaluation threats summarized in Section 5, the approach has several method-level limitations. First, hypothesis-

Table 10: Per-application distinct recall (simulated placeholders; to be replaced by empirical results). For application a, let $\mathcal{G}_{a,t}$ be the set of canonicalized ground-truth entities of type t and $\mathcal{S}_{a,t}$ the system output set (same canonicalization) aggregated across the selected databases for the application. Per-type distinct recall is $R_{a,t} = |\mathcal{G}_{a,t} \cap \mathcal{S}_{a,t}| / |\mathcal{G}_{a,t}|$. We define **All PII** recall as $R_{a,\text{all}} = |\bigcup_t \mathcal{G}_{a,t} \cap \bigcup_t \mathcal{S}_{a,t}| \big/ |\bigcup_t \mathcal{G}_{a,t}|$. A dash ("-") indicates that no ground-truth entities of the given type are present for the application (recall undefined); zero values indicate that ground-truth entities exist but none were recovered.

ID	Application	Email	Phone	User Name	Person Name	Postal Address	All PII
A1	WhatsApp	-	0.92	0.55	0.80	-	0.78
A2	Snapchat	1.00	0.88	0.30	0.60	-	0.45
A3	Telegram	-	-	-	-	-	-
A4	Google Maps	0.85	1.00	0.50	-	-	0.75
A5	Samsung Internet	0.70	-	-	-	-	0.70
I1	WhatsApp (iOS)	-	0.97	-	0.85	1.00	0.90
I2	Contacts	0.80	0.98	-	0.82	-	0.93
I3	Apple Messages	0.00	0.75	-	-	-	0.60
I4	Safari	-	-	-	-	-	-
I5	Calendar	1.00	-	-	1.00	-	1.00

driven search introduces sensitivity to planning and prioritization decisions: if early probes do not reveal strong signals for a relevant region, the planner may allocate the remaining budget elsewhere, which can reduce discovery for low-salience evidence. Second, the method relies on confidence-based escalation and stopping decisions; miscalibrated confidence can lead to premature termination or unnecessary escalation, affecting both evidentiary yield and efficiency. Finally, the current system focuses on per-type PII discovery and does not yet perform explicit cross-entity or cross-application linking (e.g., associating identifiers with accounts or correlating identities across apps), which limits downstream attribution and relationship analysis. Addressing these limitations via improved calibration and adaptive probing is promising future work but is outside the scope of this paper.

Phone 10xxx000 why fake numbers. why?

7 Conclusion and Future Work

We present a hypothesis-driven approach for discovering PII in heterogeneous mobile forensic databases and show that it can surface diverse PII while substantially reducing the effective search space. Future work includes extending discovery to additional PII categories (e.g., financial identifiers and location traces), developing hybrid strategies that combine hypothesis-driven search with selective validation, incorporating analyst-in-the-loop guidance for hypothesis refinement, and improving confidence calibration and stopping criteria for principled termination under bounded budgets. For reproducibility, we have open-sourced the framework and evaluation datasets, and they can be accessed at https://github.com/xxx/mobile-pii-discovery-agent

Acknowledgments

...

Ethical Considerations

Within up to one page, explain the ethical considerations of your work. This appendix must have exactly this title, otherwise you will risk desk rejection. Carefully study the Ethics Guidelines before submitting your paper.

Open Science

Within up to one page, this appendix must list all artifacts necessary to evaluate the contribution of the paper and make clear how the review committees can access each artifact. This appendix must have exactly this title, otherwise you will risk desk rejection.

Table 11: Distribution of distinct PII entities discovered across evaluated methods and PII categories. Benchmark (BM) values represent general LLM performance as measured by MMLU accuracy; non-LLM baselines do not have BM scores. We additionally report corpus-level distinct precision and recall (defined via set overlap after canonicalization).

Method/LLM		**Email**	**Phone**	**User Name**	**Real Name**	**Precision**	**Recall**
Name	**BM**						
bulk_extractor-v1.6	–	6046	1929	0	0	13.03%	16.79%
spaCy	–	xx	xx	0	0	13.03%	16.79%
GPT-4o-mini	82.0%	1227	1041	1685	994	39.44%	31.53%
Gemini-2.5-Pro	88.4%	3	250	226	470	84.09%	12.90%
Qwen2.5-72B	82.3%	3317	1744	7749	2233	12.99%	31.58%
LLaMA-3.1-70B-Instruct	xx	xx	xx	xx	xx	xx	xx
LLaMA-3.1-8B-Instruct	xx	xx	xx	xx	xx	xx	xx
Mixtral-8x22B	xx	xx	xx	xx	xx	xx	xx
Mixtral-8x7B	xx	xx	xx	xx	xx	xx	xx
Mistral-Large	xx	xx	xx	xx	xx	xx	xx
GPT-5.1	xx	xx	xx	xx	xx	xx	xx
GPT-4.1	xx	xx	xx	xx	xx	xx	xx
GPT-3.5-turbo	xx	xx	xx	xx	xx	xx	xx

Table 12: Effect of method choice on PII search efficiency. Values are averaged across applications. Columns examined are measured over candidate columns in the selected databases for each application; search space reduction is computed relative to scanning all candidate columns.

Method/LLM	**Avg. Cols Examined**	**Avg. Search Space Reduc.**
bulk_extractor-v1.6 (baseline)	NA	0.0%
GPT-4o-mini	18.4	92.6%
Gemini-2.5-Pro	20.3	91.1%
Qwen2.5-72B	17.6	93.4%
LLaMA-3.1-70B-Instruct	xx	xx
LLaMA-3.1-8B-Instruct	xx	xx
Mixtral-8x22B	xx	xx
Mixtral-8x7B	xx	xx
Mistral-Large	xx	xx
GPT-5.1	xx	xx
GPT-4.1	xx	xx
GPT-3.5-turbo	xx	xx

References

[1] Daniel Addai, Sarfraz Shaikh, Eric Xu, Wenbin Zhang, and Weifeng Xu. A graph-based approach for discovering evidence relationships across multiple devices in group crimes. In *2024 IEEE 24th International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*, pages 1312–1313. IEEE, 2024.

[2] Yang An, Baizhou Cheng, Chen Chen, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL: https://arxiv.org/abs/2412.15115.

[3] Claudio Anglano, Marco Canonico, and Matteo Guazzone. Forensic analysis of Telegram Messenger on Android smartphones. *Digital Investigation*, 23:31–49, 2017. doi:10.1016/j.diin.2017.09.002.

[4] Cosimo Anglano. Forensic analysis of whatsapp messenger on android smartphones. *Digital Investigation*, 11(3):201–213, 2014. Also available as arXiv:1507.07739. doi:10.1016/j.diin.2014.04.003.

[5] Chao-Chun Cheng, Chen Shi, Neil Z. Gong, and Yiran Guan. EviHunter: Identifying Digital Evidence in the Permanent Storage of Android Devices via Static Analysis. In *Proc. of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1338–1350. ACM, 2018. doi:10.1145/3243734.3243808.

[6] Eman Daraghmi, Zaer Qaroush, Monia Hamdi, and Omar Cheikhrouhou. Forensic operations for recognizing sqlite content (forc): An automated forensic tool for efficient sqlite evidence extraction on android devices. *Applied Sciences*, 13(19):10736, 2023.

[7] Simson L. Garfinkel. Digital media triage with bulk data analysis and bulk_extractor. *Computers & Security*, 32:56–72, 2013. doi:10.1016/j.cose.2012.09.011.

[8] Google DeepMind. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. URL: https://arxiv.org/abs/2507.06261.

[9] Jeel Piyushkumar Khatiwala, Daniel Kwaku Ntiamoah Addai, and Weifeng Xu. Evaluating the reliability of digital forensic evidence discovered by large language model: A case study. In *The 49th Annual International Conference on Computer Software and Applications (COMPSAC 2025)*, Toronto, Canada, July 2025. IEEE. (Acceptance rates: 27%).

[10] Xiaodong Lin, Ting Chen, Tong Zhu, Kun Yang, and Fengguo Wei. Automated Forensic Analysis of Mobile Applications on Android Devices. *Digital Investigation*, 26 (Supplement):S59–S66, 2018. doi:10.1016/j.diin.2018.04.011.

[11] Yupei Liu, Yuqi Jia, Jinyuan Jia, and Neil Zhenqiang Gong. Evaluating {LLM-based} personal information extraction and countermeasures. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 1669–1688, 2025.

[12] Kunal Mukherjee and Murat Kantarcioglu. Llm-driven provenance forensics for threat investigation and detection. *arXiv preprint arXiv:2508.21323*, 2025. URL: https://arxiv.org/abs/2508.21323.

[13] OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. OpenAI Blog, 2024. Accessed: 2026-01-13. URL: https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

[14] Kevin Pagano. The evidence locker: A dfir image compendium. https://theevidencelocker.github.io/, 2025. Accessed: 2024-12-21.

[15] Darren Quick and Kim-Kwang Raymond Choo. Data volume forensics: Challenging the high volume data analysis limitation of digital forensic investigations. *Digital Investigation*, 11(1):22–30, 2014.

[16] B. Sharma, J. Ghawaly, K. McCleary, A. M. Webb, and I. Baggili. Forensicllm: A local large language model for digital forensics. *Forensic Science International: Digital Investigation*, 52, 2025. doi:10.1016/j.fsidi.2025.301872.

[17] Akila Wickramasekara and Mark Scanlon. A Framework for Integrated Digital Forensic Investigation Employing AutoGen AI Agents. In *Proc. of the 12th International Symposium on Digital Forensics and Security (ISDFS)*. IEEE, 2024. doi:10.1109/ISDFS60797.2024.10527291.

[18] Ruoyao Xiao, Yu Luo, Weifeng Xu, and Dianxiang Xu. From text to stix: Reducing hallucinations through fine-tuned llms. In *2025 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2025. Guiyang, China.

[19] Eric Xu, Wenbin Zhang, and Weifeng Xu. Transforming digital forensics with large language models: Unlocking automation, insights, and justice. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 5543–5546, Boise, USA, October 2024.

[20] Weifeng Xu and Dianxiang Xu. Visualizing and reasoning about presentable digital forensic evidence with knowledge graphs. In *The 19th Annual International Conference on Privacy, Security and Trust (PST2022)*, pages 1–10, Fredericton, Canada, August 2022. IEEE.

[21] Weifeng Xu, Jie Yan, and Hongmei Chi. A forensic evidence acquisition model for data leakage attacks. In *The 17th IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 53–58, Shenzhen, China, July 2019. IEEE.

[22] Honghe Zhou, Weifeng Xu, Josh Dehlinger, Suranjan Chakraborty, and Lin Deng. An llm-based approach to gain cybercrime insights with evidence networks. In *Proceedings of the 20th Symposium on Usable Privacy and Security (SOUP 2024)*, Philadelphia, PA, August 11-13 2024.

Table 13: PII Per Application and Database (Ground Truth)

ID	Application	Database	Email	Phone	User Name	Person Name	Postal Address	Total
A1	WhatsApp	wa	0	16	180	16	0	**212**
		msgstore	0	1	0	0	0	**1**
		commerce	0	0	0	0	0	**0**
A2	Snapchat	core	1	1	1	1	0	**4**
		main	0	14	4091	15	0	**4120**
		journal	0	0	0	0	0	**0**
A3	Telegram	cache4	0	0	0	0	0	**0**
		cache4	0	0	0	0	0	**0**
		cache4	0	0	0	0	0	**0**
A4	Google Maps	gmm_storage	0	0	0	0	0	**0**
		gmm_myplaces	0	0	0	0	0	**0**
		peopleCache	2	1	3	0	0	**6**
A5	Samsung Internet	SBrowser	1	0	0	0	0	**1**
		SBrowser2	0	0	0	0	0	**0**
		searchengine	0	0	0	0	0	**0**
I1	WhatsApp	ChatStorage	0	1	0	53	2	**56**
		ContactsV2	0	1014	0	1008	0	**2022**
		CallHistory	0	0	0	0	0	**0**
I2	Contacts	AddressBook	6	1013	0	1008	0	**2027**
		AddressBookImages	0	0	0	0	0	**0**
I3	Apple Messages	sms	7	30	0	0	0	**37**
I4	Safari	History	0	0	0	0	0	**0**
		CloudTabs	0	0	0	0	0	**0**
I5	Calendar	Calendar	1	0	0	1	0	**2**
		Extras	0	0	0	0	0	**0**

Table 14: PII configuration shared by exploration and targeted extraction. The `regex` is used to prefilter sampled rows/values during exploration, while `desc` is injected into LLM prompts in both exploration (classification/validation) and extraction (type-aware entity recovery).

PII key	**Type label**	**Regex (prefilter)**	**LLM description (`desc`)**
`EMAIL`	email address	`[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-] +\.[a-zA-Z]{2,}`	a unique identifier for a destination to which electronic mail (email) can be sent and received over the internet; examples include jane.doe@example.com, john.smith@provider.net, dev-team@startup.io, and user.name+label@domain.org
`PHONE`	US phone number	`\+?[0-9]{1,4}[- .]?\(?[0-9]{1,3}?\)?[- .]?[0-9]{1,4}[- .]?[0-9]{1,4}[- .]?[0-9]{1,9}`	a US phone number is a 10-digit NANP number (area code + exchange + line) that may be written as 2023133725, 202-313-3725, (202) 313-3725, 202.313.3725, +1 202 313 3725, or 1-202-313-3725
`USERNAME`	username	`\b[a-zA-Z][a-zA-Z0-9._]{2,51}\b`	a unique or quasi-unique identifier used to authenticate or reference a user account on an application or online service. Even if pseudonymous, it is treated as PII when it can be linked to a specific individual through account records, cross-app correlation, or supporting metadata. Examples include John.smith, ericxu99, marsha_mellos, heisenberg-ercarro, x7Qp_13, user_482019
`PERSON_NAME`	person name	`[A-Za-z][A-Za-z\s\.\-]{1,50}`	a loosely structured human name-like strings that typically consist of a first name, a first name and a last name, and may also include middle names, initials, prefixes (e.g., Mr., Dr.), and suffixes (e.g., Jr., Sr.)
`POSTAL_ADDRESS`	US postal address	`(?i)\b(?:p\.?\s*o\.?\s*box|post \s+office\s+box|ave \.?|avenue|st\.?|street|rd \.?|road|blvd\.?|boulevard|dr \.?|drive|ln\.?|lane|ct\.?|court|pl \.?|place|way|pkwy textbackslash .?|parkway|cir \.?|circle|ter\.?|terrace|hwy \.?|highway|trl\.?|trail|sq \.?|square|pike|loop|run|walk|path|byp \.?|bypass|(?:n|s|e|w|ne|nw|se|sw) \b) \b`	a US postal address is a street-level mailing location in the United States, commonly appearing as a street name and suffix (e.g., 'Market St') optionally with a street number (e.g., '1500 Market St'), unit, city/state, ZIP, or a PO Box (e.g., 'P.O. Box 123')

Table 15: Prompt inventory used in the empirical study. Prompts are grouped by pipeline stage. In exploration, `regex` prefilters sampled rows/values and `desc` is injected into the LLM prompt for type-aware classification. In targeted extraction, `desc` is also injected to guide normalization and filtering of extracted PII instances.

ID	**Stage**	**Message role**	**Purpose and key injected fields**	**Expected output / constraints**
P1	Exploration (SQL synthesis)	System	Generate an exploratory SQLite query to sample candidate rows/values for a target PII type. Injects: `type` label and `regex` (prefilter). Encourages use of schema tools (e.g., list tables, inspect column types) and selection of likely columns.	Single SQL query (no markdown). Uses `REGEXP` prefilter and sampling limit (e.g., `LIMIT k`); avoids destructive statements; returns enough context to support downstream classification (e.g., table/column/value snippets, row identifiers when feasible).
P2	Exploration (agent guardrails)	System	Forensic analyst role and safety/format guardrails for the SQL-generation agent. Emphasizes SQLite-only context, careful reasoning about schemas, and tool-usage discipline.	Outputs only the requested artifact (SQL or JSON per tool), no extra prose; avoids `UPDATE/DELETE/DROP`; prefers read-only queries; keeps queries syntactically valid for SQLite.
P3	Exploration (sample classification)	System + User	Classify whether the **prefiltered sample** contains the target PII type. Injects: `desc` (type description) and `type` label. User provides sampled rows/values (often as JSON snippets).	Strict JSON object with fields like: {`found`: bool, `confidence`: float in [0,1], `evidence`: list of short supporting strings}. No additional keys/text.
P4	Targeted extraction (normalization)	System + User	Extract *real* instances of the target PII from candidate text/values returned by prefiltering and/or schema-guided selection. Injects: `desc` (type description) and `type` label; instructs filtering false positives and producing canonicalizable instances.	Strict JSON array of strings (the extracted PII instances). No extra prose. Values should be ready for downstream canonicalization/deduplication (e.g., trimmed emails, digit-normalized phones when applicable).

Table 16: Exact LLM prompts (verbatim). Braced tokens denote runtime-injected fields.

ID	Role	Prompt text
P1	System	You are a forensic analyst working with a single SQLite database. Task: generate ONE read-only SQLite SELECT query that samples candidate rows/values likely to contain the target PII type. Target PII type label: {type} Prefilter regex (SQLite REGEXP): {regex} Requirements: - Output ONLY the SQL query (no markdown, no explanation). - Use REGEXP to prefilter candidate values using {regex}. - Use LIMIT {k} (or equivalent) to bound results. - Do NOT use UPDATE/DELETE/INSERT/DROP/ALTER/CREATE. - Prefer returning table/column/row identifier and matched value. - Keep the query valid SQLite.
P2	System	Guardrails: - Output only the requested artifact (SQL or JSON), with no prose. - SQLite only. - Read-only queries only. Never output UPDATE/DELETE/INSERT/DROP/ALTER/CREATE. - Prefer minimal SQL likely to execute. - If uncertain, sample small results.
P3	System	You are a forensic analyst. Determine whether the provided prefiltered sample contains instances of the target PII type. Definition (desc): {desc} Type label: {type} Return a STRICT JSON object with exactly these keys: - found: boolean - confidence: number in [0,1] - evidence: array of short strings No additional keys. No surrounding text.
P3	User	Target type: {type} Definition: {desc} Prefiltered sample (JSON): {sample} Return the strict JSON object only.
P4	System	You are a forensic analyst. Extract real instances of the target PII type from the provided candidate strings/rows. Type label: {type} Definition (desc): {desc} Return a STRICT JSON array. Each element must be an object with exactly: - value - normalized - source Rules: - Filter false positives aggressively. - Do not invent values. - No additional keys. No surrounding text.
P4	User	Target type: {type} Definition: {desc} Candidates: {candidates} Return the strict JSON array only.