Large-scale online deanonymization with LLMs

Simon Lermen* *MATS*
Daniel Paleka* *ETH Zurich*
Joshua Swanson *ETH Zurich*
Michael Aerni *ETH Zurich*
Nicholas Carlini *Anthropic*
Florian Tramèr *ETH Zurich*

Abstract

We show that large language models can be used to perform at-scale deanonymization. With full Internet access, our agent can re-identify Hacker News users and Anthropic Interviewer participants at high precision, given pseudonymous online profiles and conversations alone, matching what would take hours for a dedicated human investigator. We then design attacks for the closed-world setting. Given two databases of pseudonymous individuals, each containing unstructured text written by or about that individual, we implement a scalable attack pipeline that uses LLMs to: (1) extract identity-relevant features, (2) search for candidate matches via semantic embeddings, and (3) reason over top candidates to verify matches and reduce false positives. Compared to classical deanonymization work (e.g., on the Netflix prize) that required structured data , our approach works directly on raw user content across arbitrary platforms. We construct three datasets with known ground-truth data to evaluate our attacks. The first links Hacker News to LinkedIn profiles, using cross-platform references that appear in the profiles. Our second dataset matches users across Reddit movie discussion communities; and the third splits a single user's Reddit history in time to create two pseudonymous profiles to be matched. In each setting, LLM-based methods substantially outperform classical baselines, achieving up to 68% recall at 90% precision compared to near 0% for the best non-LLM method. Our results show that the practical obscurity protecting pseudonymous users online no longer holds and that threat models for online privacy need to be reconsidered.

1 Introduction

For decades, it has been known that individuals can be uniquely identified from surprisingly few attributes. Sweeney's seminal work demonstrated that 87% of the U.S. population could be uniquely identified by just zip code, birth date, and gender [34]. Narayanan and Shmatikov showed that anonymous Netflix ratings could be linked to public IMDb profiles using only a handful of movie preferences [24], while De Montjoye et al. [6] proved that four spatiotemporal points are enough to uniquely identify 95% of individuals in mobile phone datasets. Despite these attacks, pseudonymous online accounts (Reddit throwaways, anonymous forums, review profiles, etc) have remained largely unaffected by deanonymization attempts. The reason is simple: applying such attacks in practice has required structured data amenable to algorithmic matching or substantial manual effort by skilled investigators reserved for high-value targets [13].

Deanonymization is a two-step process at heart, involving profiling an anonymous person from their posts, and then matching them to a known identity. It's well-known that large language models (LLMs) can infer personal attributes from text on online forums [8, 29, 38]. Given this, it makes sense to ask: how good are LLMs at full end-to-end deanonymization, and is this a practical threat to pseudonymous accounts?

Our contributions. We demonstrate that LLMs fundamentally change the picture, enabling fully automated deanonymization attacks that operate on unstructured text at scale. We show this by phrasing deanonymization as a matching problem and showing LLMs can perform all steps needed to match accounts: extract identity-relevant signals from arbitrary text, efficiently search over millions of candidate profiles, and reason about whether two accounts belong to the same person. We show that the practical obscurity that has long protected pseudonymous users (the assumption that deanonymization, while theoretically possible, is too costly to execute broadly) no longer holds.

We validate this thesis in three deanonymization settings: (1) matching an online account to its real identity; (2) linking an identity to an unknown pseudonymous account; and (3) linking pseudonymous accounts of the same person across different platforms or time periods. These settings capture distinct threat models (e.g., doxxing of an online account, a stalker targeting a victim, or an adversary consolidating a user's activity) and pose different technical challenges.

*Equal contribution.

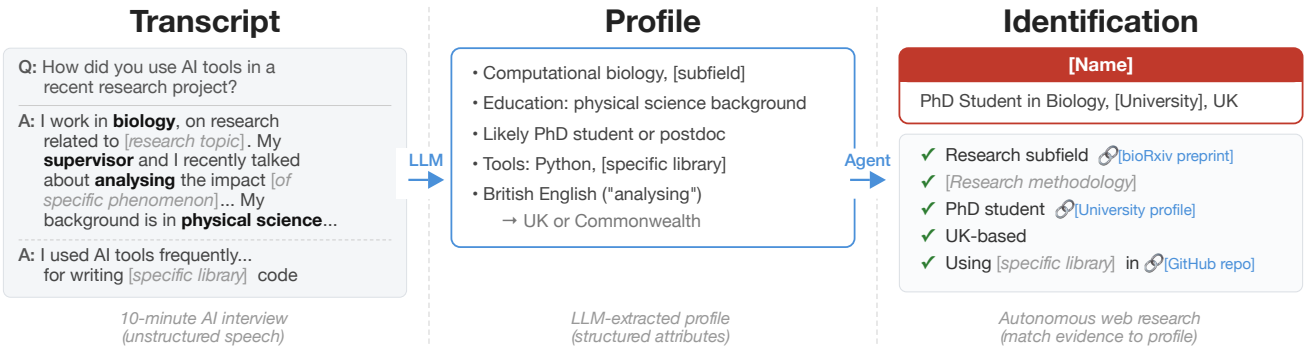


Figure 1: End-to-end deanonymization from a single interview transcript from [2] (*details altered to protect the subject's identity*). An LLM agent extracts structured identity signals from a conversation, autonomously searches the web to identify a candidate individual, and verifies the candidate matches all extracted claims.

In the first setting, we demonstrate that state-of-the-art LLM agents already possess reasonable capabilities to perform end-to-end deanonymization fully autonomously on the open web. This is the most challenging setup we consider: Given only an anonymous online profile, our LLM agents autonomously search the web, query databases, enumerate potential candidate identities, and reason over evidence to identify who the profile belongs to. In a study of Hacker News and Reddit profiles, these agents achieve 25–67% recall with 70–90% precision, replicating in minutes what could take hours for a dedicated human investigator. Figure 1 illustrates a successful deanonymization of a participant in the Anthropic Interviewer Dataset [2, 21], a collection of discussions with 125 scientists about how they integrate AI into their work. We estimate that our agents correctly re-identify at least 9/125 participants from this dataset. *This capability comes for free*: we simply prompt frontier agents with a summary of an online profile and ask the agent to uncover the identity behind it.

To enable a more granular study of how LLMs can improve upon prior deanonymization attacks in the second and third settings (linking pseudonymous accounts to an identity or across platforms), we then design a scalable deanonymization pipeline in four LLM-augmented stages:

- *Extract*: as in prior work [8, 29], we ask LLMs to identify and structure relevant features from unstructured posts: demographics, writing style, interests, incidental disclosures, etc.
- *Search*: we encode extracted features into dense embeddings enabling efficient search over thousands or millions of candidate profiles.
- *Reason*: we use extended reasoning on top candidates from the search step to identify the most likely match given all available context;
- *Calibrate*: we prompt LLMs to provide confidences in identified matches (either absolute or relative to other matches), which lets us calibrate the attack to a desired false positive rate.

Our pipeline substantially outperforms adaptations of classical deanonymization techniques [24]. For example, we improve recall from 0.1% to 45.1% at 99% precision when linking Hacker News accounts to LinkedIn profiles; and we improve recall from 0% to 2.8% at the same precision threshold on a challenging task of linking Reddit users across movie subreddits. Ablations confirm that each pipeline stage contributes: in particular, the *Reason* step improves recall at 99% precision from 4.4% (Search only) to 45.1%.

Our work introduces an evaluation framework for large-scale deanonymization attacks, which we believe can be broadly useful for follow-up studies. Indeed, evaluating deanonymization attacks at scale poses inherent challenges, since ground-truth labels are difficult to obtain without compromising the privacy of real users. As a result, previous work evaluated scale through synthetic data [25] and relied on manual verification or guess-work to validate attacks on real data [5, 24]. We propose two approaches that balance real-world validity with research ethics. Our first approach identifies profiles that are *not* anonymous (for instance, a Hacker News account whose "about" field links to a LinkedIn profile) and then renders them pseudonymous for evaluation purposes by removing all direct identifiers. We then measure whether our attacks can recover the removed link. Although this approach may not capture the behavior of the most privacy-conscious users, it provides verifiable ground truth at scale and enables rigorous comparison across methods. Our second approach splits a single user's activity across communities or across time, synthetically creating two user profiles with a known link which our attacks then try to infer. This mitigates selection bias concerns but comes with other limitations discussed later.

Implications. Our findings have significant implications for online privacy. The average online user has long oper-

ated under an implicit threat model where they have assumed pseudonymity provides adequate protection because targeted deanonymization would require extensive effort. LLMs invalidate this assumption. As also observed by Staab et al. [29], they do not necessarily do so by exceeding human *capability*—the signals our models exploit are the same signals that a skilled investigator would recognize—but by reducing *cost*. Our experiments provide quantitative evidence for this in the deanonymization setting. We discuss our relationship to related work on online profiling and privacy in Section 7; and address memorization concerns in Section 8.

Past research identified that generally the only effective mitigation against privacy attacks applied to "anonymized" structured data was simply not to release such data at all [24]. Our findings suggest that unstructured text data deserves similar reconsideration. However, this text is the very content that makes online communities valuable. We therefore argue that privacy expectations, platform policies, and social norms that govern pseudonymous participation online require reconsideration. We hope that this work can spark that conversation.

2 LLM agents can autonomously re-identify anonymous online profiles

To best highlight the paradigm shift that LLMs bring in online deanonymization, we begin our study by directly evaluating whether frontier LLMs can autonomously perform end-to-end deanonymization—from anonymous profile to real-world identity. This setting has so far required a significant manual component even if LLMs are used for attribute extraction [8].

In order to evaluate whether LLMs can carry out this capability, we first need to build an evaluation set of online profiles along with a known ground truth identity. We do this by collecting *non-anonymous* online profiles and then using an LLM to remove any directly identifying information. Table 2 summarizes the anonymization rules we use: we remove personal URLs, social media handles, GitHub repositories (which directly identify the owner) and retain institutions, demographics, interests, and names of connected people (which are too broad to identify the profile owner alone). The goal is to create a task that would be nontrivial for a motivated human. We then test whether LLM agents can solve this task by performing web searches. This artificial anonymization is only needed for evaluation (to establish ground truth); any real attack would target profiles that are already pseudonymous.

We provide the agent with a text description of a person (derived from their pseudonymous posts) and ask it to determine their real identity by searching the web, cross-referencing sources, and reasoning over evidence.

Our pipeline proceeds as follows:

1. Summarize the user's posts into a profile containing both stated and inferred facts about demographics, career history, interests, and so on.
2. Generate a search prompt from the profile. Anonymize the search prompt by removing names, handles, and unique identifiers that would allow for a direct search.
3. Pass the anonymized search prompt to an LLM agent with web search tools. The agent autonomously searches, reasons over findings, and attempts to identify the user.

Following Li [21], to prevent misuse, we describe our attack at a high level, and do not publish the agent, exact prompts, or tool configurations used. Running the agent on each profile costs us between $1–$4; making the cost of the experiments below less than $2000.

2.1 Results

We first evaluate our agentic pipeline on three ground-truth datasets based on Hacker News and Reddit profiles. In the interest of research ethics, **we do not evaluate our method on any truly pseudonymous accounts on Hacker News and Reddit**. We additionally report on the deanonymization of genuinely pseudonymous Anthropic Interviewer transcripts that were the subject of prior deanonymization research [21].

Hacker News → LinkedIn. We collect 338 Hacker News (HN) users who linked a LinkedIn profile in their publicly-visible HN bio, providing verified real-world identities as ground truth. We first summarize each user's HN activity (comments and stories) into a structured profile. Then we create a search prompt and anonymize it (see Appendix A for details), and pass it to the agent. The agent correctly identifies 226 of 338 targets (67%) at 90% precision (95% CI: 86–93%; 25 incorrect identifications, 86 abstentions). We note that these edited profiles are much easier to identify than most pseudonymous accounts; we discuss this bias in Section 3.3.

We also evaluate the agent on two smaller Reddit datasets, academic users and career-discussion posters, where it achieves 25–52% recall at 72–90% precision; we report the full details in Appendix B.

Anthropic Interviewer transcripts. Anthropic publicly released transcripts of 125 AI-conducted interviews with scientists [2, 16], in which participants described their use of AI in their research. The transcripts were partially redacted for participant privacy. Li [21] showed that LLM agents with web search can match project descriptions in these transcripts to published papers; specifically, they use a task decomposition approach [18] to bypass safeguards by breaking the attack down into individual benign tasks. They eventually ran their attack on 24 scientists conversations and recovered 6 identities. We attack the same dataset with our pipeline, *without task decomposition*. Concretely, we construct profiles from transcripts in a similar way as we do for Hacker News and Reddit users; and then run the agentic search on 33 scientists that talked about their past research in the transcripts.

The agent correctly identifies 9 out of 33 scientists at 82% precision (2 wrong, 22 refusals or abstentions). Here, we are uncertain about the true deanonymization rate, as we have to manually verify the matches. The agent claims to identify 11 scientists; but for 2 out of 11 we notice the transcript contradicts the identified academic profile in some ways. We skip the anonymization step in this experiment: the transcripts by design do not contain platform handles or usernames, and we report deanonymization judged by a human, following the precedent set by Li [21]. Our judgment is that our agent finds 50% more matches than Li [21], but the exact accuracy in either of these experiments cannot be fully verified.

Limitations of the agentic approach. The agentic approach demonstrates that LLM-based deanonymization is a real threat, but it limits the science we can do. First, ground-truth datasets are small because they require verified identity links. Second, in privacy research the standard metric is a precision-recall curve; but our approach produces a single point estimate of precision and recall rather than a tunable tradeoff. Finally, the attack relies on opaque web search systems, making it difficult to isolate what the LLM agent contributes versus what the search engine embeddings contribute.

3 A framework for scalable deanonymization with LLMs

To better understand how LLMs deanonymize people, we need to explicitly state the different steps of successful deanonymization For this reason, we introduce a modular framework that decomposes deanonymization into distinct stages, each of which can be run with or without an LLM. While non-agentic, this approach lets us scale to larger datasets, trace precision-recall curves, and ablate the contribution of LLMs at each step. Our framework is inspired by the seminal work of Narayanan and Shmatikov [24], whose attack serves as a "classical" baseline that we systematically augment with LLM components.

We first define our threat model and the ESRC deanonymization framework in Section 3.1 and Section 3.2, respectively. In Section 3.3, we then introduce an evaluation procedure to quantify the effectiveness of LLM-based deanonymization for different settings (identifying pseudonymous accounts from known identities in Section 4 and matching pseudonymous profiles in Sections 5 to 6).

3.1 Threat model

We build on [24], who introduce large-scale deanonymization attacks by reconstructing *micro-data*. Micro-data is information at the level of an individual, such as "gave Twilight a 5-star rating", "lives in Texas", or "never capitalizes sentences". This information alone may not be identifying, but it can identify a pseudonymous account by *matching* their micro-data against a database of micro-data with known identities. Narayanan and Shmatikov [24] (hereafter termed the "Netflix Prize attack") demonstrate this by matching anonymized Netflix Prize ratings to public IMDb profiles based on movie rating micro-data.

Attacker goal. Our attacker's goal is to deanonymize pseudonymous online identities, which we model as the goal of matching profiles belonging to the same user across two sets. Concretely, our attacker is given a *query user* profile and a large set of *candidate user* profiles. Given those inputs, the attacker either returns a best-guess match of the query user in the candidate set or abstains. The attacker's goal is to produce a correct guess if the query user has a corresponding candidate, and it should abstain if there is no matching candidate.

Controlling deanonymization difficulty. The difficulty of finding a match depends on two key factors: the number of candidates and the prior probability that the query user has a matching candidate. Larger candidate pools obviously make the attack harder: given just two candidates (one of which is a correct match), even random guessing correctly matches about half of all queries. In contrast, identifying the correct user in 10k candidates requires a much stronger attacker. The magnitude of the prior probability of a match influences the attacker's precision (i.e., if the prior is very low, then any non-abstaining guess is likely false *a priori*).

For most of this paper, we will fix these parameters by using a fixed candidate pool for each setting and assuming a best-case scenario where every query user has a true match in the candidate set (i.e., the prior of a match existing is 100%). For full generality, we perform a detailed analysis in which we vary these two difficulty parameters in Section 6.4.

3.2 The ESRC deanonymization framework

Our deanonymization framework follows the structure of the seminal attack by Narayanan and Shmatikov [24], augmented with LLM components. Figure 2 illustrates how LLMs make each part of the pipeline easier. Concretely, all our experiments involve the following steps:

1. **Extract**: The attack first needs to extract micro-data for query and candidate users. The original Netflix Prize attack directly receives structured micro-data in the form of Netflix and IMDb rating vectors as input. For arbitrary online profiles, we use LLMs to extract semi-structured summaries from unstructured posts and comments. As a result, the LLM-augmented attack may be able to rely on much more information.

2. **Search**: Given micro-data for the query and candidate users, the second stage of the attack searches for the most likely matches. In the Netflix Prize attack, this is done with a simple (weighted) vector similarity search

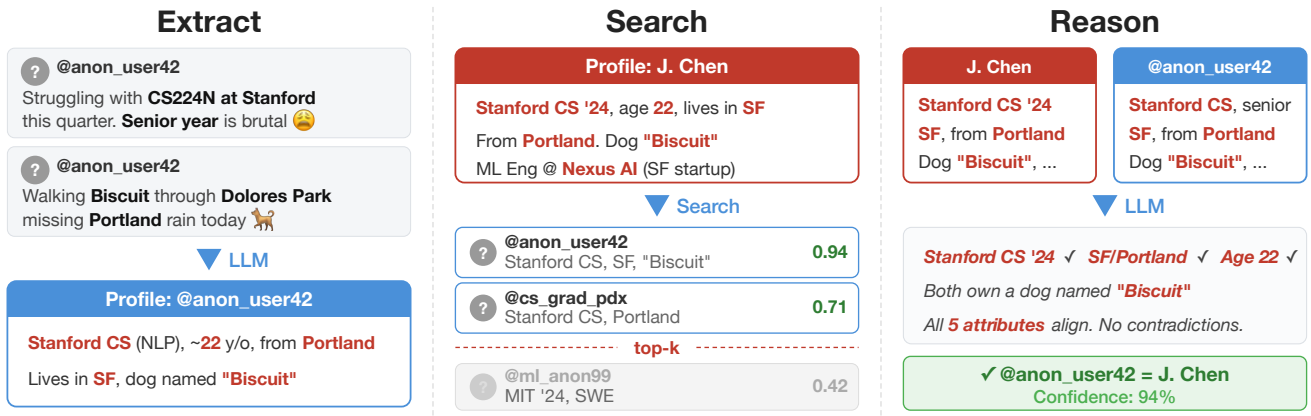


Figure 2: Overview of our framework for large-scale deanonymization. Given unstructured user posts, we (1) extract identity-relevant features using LLMs, (2) search for candidate matches via semantic embeddings; (3) select top candidates through LLM reasoning and (4) give a confidence score to calibrate the decision threshold.

over feature vectors (i.e., the candidate with the highest vector similarity is returned). For our method, we do this by performing a nearest-neighbor search over *LLM embeddings* of the extracted summaries.

3. **Reason:** LLMs allow us to select from a shortlist of multiple top matches returned by the Search step, whereas the Netflix Prize attack could only return the single best match by vector similarity. Additionally, we can first use a cheaper LLM to select the most likely candidate, then use a more expensive model with advanced reasoning capabilities to verify the selection.

4. **Calibrate:** Finally, we need to calibrate the tradeoff between precision (how many of our predictions are correct) and recall (how many true matches we identify). The Netflix Prize attack does this by thresholding the similarity score of the top match to control the attack's confidence (i.e., if the top match's similarity is above a given threshold, the attacker guesses this match, and otherwise abstains). Similarly, we can use the embedding similarity score from the Search step, or an LLM-produced confidence score from the Reason step, or ratings computed by sorting matches via pairwise LLM comparisons. We use these confidence scores to trace precision-recall curves.

3.3 Experimental Setup

To enable quantitative evaluations of the benefits of LLMs at scale, we instantiate our deanonymization framework in settings with verifiable ground truth. To obtain this ground truth, we consider two approaches:

1. *Synthetically anonymized matches.* We search for user profiles that explicitly link themselves to other platforms (e.g., a Hacker News user that posts their LinkedIn profile). Since these profiles trivially expose links, we carefully sanitize them by removing all identifying information. Text sanitization is a well-studied problem [23, 30], though recent work shows that even sophisticated methods leave semantic signals that enable re-identification [38]; the point of our attacks is to exploit precisely the signals that mimic the traces that a genuinely anonymous user would leave in their profile.

2. *Semantic splits for arbitrary profiles.* In practice, we found only few users who explicitly link their pseudonymous account to other platforms. Hence, as a second option, we split a single user's profile into two semantically distinct parts (e.g., temporally). Those splits provide the strongest source of ground truth information at the cost of being less realistic.

Any deanonymization setup with ground truth introduces distributional biases. In our cross-platform datasets, the profiles are likely easier to deanonymize than an average profile: the very fact that ground truth exists implies that the user may not have cared about anonymity in the first place. Similarly, two split-profiles of a single user are inherently alike, whereas two pseudonymous accounts of the same person (e.g., an official and a pseudonymous alt account) might expose more heterogeneous micro-data.

We thus focus on methods with very few false positives. Due to the aforementioned biases, our evaluation on ground truth settings might overestimate an attack's recall (how many users are successfully deanonymized). But we argue that the false positive rate (i.e., the likelihood of a wrong guess instead of abstaining) does transfer to real-world attacks; there is no reason why a well-calibrated attacker should make more wrong guesses on the average internet profile than on our

ground truth profiles. Therefore, similarly to [24], we focus on attacks with a very low false positive rate. This ensures that whenever our attacks do return a guess, the guess is likely to be correct—independent of the setting.

Concretely, we report *Recall@Precision* for high precision values. We hence define *recall* (or *True Positive Rate (TPR)*) as the fraction of matchable users (i.e., users with a corresponding match in the candidate pool) that the attacker guesses correctly, and *precision* as the fraction of all non-abstentions that are correct. We vary the confidence threshold at which we output a match (derived from LLM embedding similarity or the Calibrate stage of the pipeline) to get a precision-recall curve.

4 Linking profiles across platforms: Hacker News and LinkedIn

We begin our evaluation with cross-platform matching between LinkedIn and Hacker News. This setting reflects a plausible real-world attack where an adversary knows a person's real identity (e.g., from LinkedIn) and seeks their pseudonymous account on a different platform.

4.1 Dataset

We collect 987 LinkedIn profiles linked to 995 Hacker News (HN) accounts (ground truth is established by users who posted their LinkedIn URL in their HN bio), drawn from a candidate pool of approximately 89,000 active HN users. Eight LinkedIn profiles are linked to multiple HN accounts that shared the same LinkedIn URL. We identified four additional HN alt accounts using strong evidence such as matching names and companies. We count a match as correct if any of the linked HN accounts is returned. Every query has a true match in the candidate set. The LinkedIn side represents the known identity with real professional profiles. The HN side serves as the anonymized target: as in Section 2, we remove names, URLs, and other direct identifiers from bios using an LLM to prevent trivial matching (see Appendix A for our complete anonymization procedure). The task is to match a LinkedIn profile with the corresponding LLM-anonymized HN account.

4.2 Attack instantiation

Extract: biographical profiles. We use an LLM to summarize each user's HN activity (comments, stories, and bio) into a biographical profile. On the LinkedIn side, we use an LLM to summarize the data that users reveal about themselves, such as job positions and school graduations.

Search: embedding similarity. We embed all 89,000 candidate profiles into a nearest-neighbor search index using Gemini embeddings [20] and FAISS [7] with cosine similarity. For each query, we embed their biographical profile and retrieve the top candidates.

Reason: two-stage selection and verification. Embedding similarity is effective at narrowing down the candidate pool to the most similar candidates, but the correct match will often not be ranked first. We therefore select the top-100 candidates by embedding similarity, then apply a two-stage reasoning process (illustrated in Figure 5 for the movie setting of Section 5). First, Grok 4.1 Fast [37] selects the most likely match from the shortlist or abstains. Then, if a candidate was selected, GPT-5.2 [27] evaluates the selected pair with either low or high reasoning effort. By only using a more capable model with advanced reasoning on a single promising match per query, we substantially reduce cost.

Calibrate: LLM confidence scores. We use the verification-stage confidence as the calibration score for precision-recall curves. For embedding-only and Narayanan baseline results, we use the gap between the top-2 candidates' similarities—a large gap indicates the top candidate stands out clearly, making it more likely correct. The model outputs a numerical confidence score as a structured field in its function-call response, and we threshold it to trade off precision and recall, as shown in Figure 3a.

Baseline. We adapt the Netflix Prize attack to professional attributes: programming languages, cities, companies, schools, job roles, and seniority levels. Each user is represented as a binary vector over 332 attributes, and we rank candidates by rarity-weighted Jaccard similarity (see Appendix C for details).

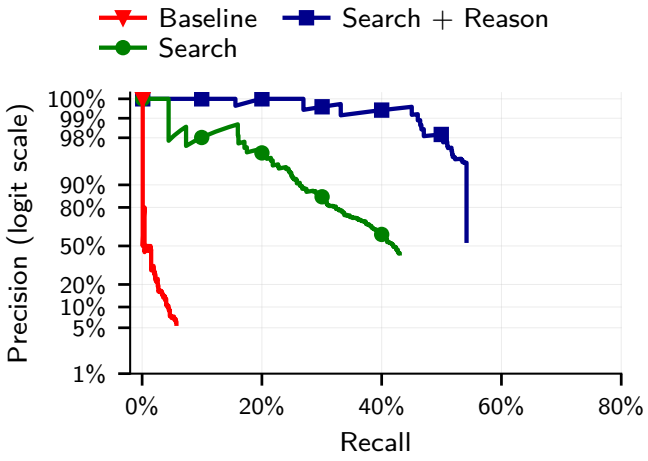
4.3 Results

LLM-based embeddings outperform handcrafted features. Figure 3a shows the precision-recall tradeoff. The embedding-based approach substantially outperforms the Netflix Prize attack on all metrics; the Netflix Prize attack correctly ranks only 5.1% of candidates as the top-1 match, and thresholding the top-2 similarity gap to improve precision quickly drives recall to near zero.

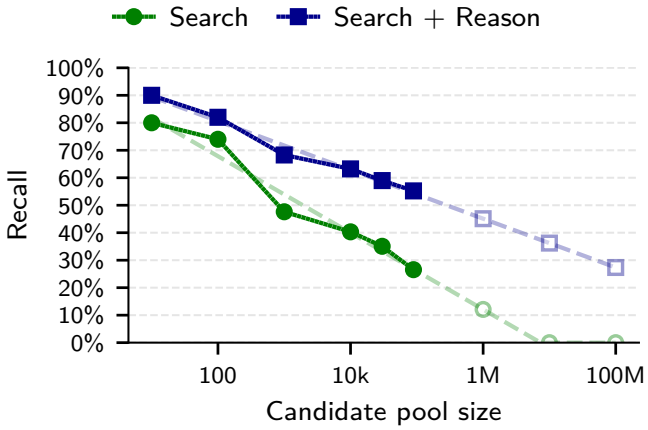
LLM reasoning further improves matching. As shown in Figure 3a, adding the Reason step substantially outperforms pure embedding search in both precision and recall. The ability of LLMs to select from a number of candidates and use reasoning complements embedding-based retrieval.

High reasoning effort helps at high precision. As shown in Table 1, high reasoning effort significantly outperforms low reasoning only at 99% precision (45.1% vs 36.0% recall). At lower precision thresholds, the two reasoning modes perform comparably.

Reasoning scales to larger candidate pools. We evaluate Search and Reason as the pool grows from 1k to 89k users.



(a) Precision-recall curves.



(b) Scaling with candidate pool size.

Figure 3: LinkedIn to Hacker News matching. (a) Precision-recall curves comparing methods: LLM-based embeddings outperform the Netflix Prize attack baseline, and LLM selection from the top-100 candidates further improves performance. (b) Matching recall @90%precision by candidate pool size. Dashed lines show log-linear extrapolation to larger pools.

Table 1: Recall at different precision thresholds for HN-LinkedIn cross-dataset user matching (987 queries). Reason uses GPT-5.2 with low or high reasoning effort. 95% Wilson confidence intervals shown.

Method	90% Prec	98% Prec	99% Prec
Narayanan baseline	0.1% (0.0–0.6)	0.1% (0.0–0.6)	0.1% (0.0–0.6)
Search (embedding)	26.3% (23.7–29.2)	16.1% (13.9–18.5)	4.4% (3.3–5.9)
Reason (low)	**55.0%** (51.9–58.0)	**44.8%** (41.8–47.9)	36.0% (33.1–39.0)
Reason (high)	**54.2%** (51.1–57.2)	**50.0%** (46.9–53.0)	**45.1%** (42.1–48.2)

As shown in Figure 3b, both methods degrade log-linearly with pool size, but Reason is substantially more robust. At 90% precision, Reason achieves 68.3% recall at 1k candidates and retains 55.2% at 89k—a loss of only 13 percentage points across nearly two orders of magnitude. Search drops more steeply, from 47.6% to 26.6% over the same range. The gap widens at high precision: at 89k candidates and 99% precision, Reason still achieves 33.0% recall while Search collapses to 4.2%. Log-linear extrapolation to 1M candidates projects Reason at approximately 45% recall (90% precision), compared to roughly 12% for Search; at 100M candidates, Reason would still retain an estimated 27% recall while Search falls to zero (see Appendix D.1 for details).

5 Linking users across Reddit communities

The original Netflix Prize attack was designed for connecting cross-platform movie ratings, not the cross-platform professional setting of Section 4. We thus construct a more direct comparison using movie-related Reddit activity. This provides a clean comparison: both methods operate on the same micro-data features (subjective movie reviews) but differ in how they represent users—we use transformer-derived text embeddings, while Narayanan and Shmatikov [24] relied on hand-crafted numerical features.

5.1 Constructing the community-split dataset

Although the Hacker News–LinkedIn setting from Section 4 demonstrates cross-platform deanonymization, its scale is limited by verifiable ground truth. Profiles for which we can confidently infer ground truth matches are rare—they require users to explicitly link accounts or directly disclose their identity on both profiles. This is a general problem when evaluating deanonymization, since the anonymity that motivates the problem also obstructs collecting ground truth data.

We solve this problem by splitting Reddit profiles: partitioning each user's activity into separate query and candidate profiles. This yields thousands of profile pairs with exact ground truth. In this section, we split by community; in Section 6, we split temporally.

This setup mimics a situation in which a user maintains pseudonymous accounts on different platforms discussing similar topics. In our case, the communities discuss either mainstream movies (r/movies) or more niche movie categories. This also helps understand the relationship between how much micro-data someone shares and how identifiable

they become. Unlike most online activity, movie discussions offer an intuitive discrete metric: the number of movies discussed and the overlap of shared movies across datasets.

Our dataset consists of movie discussion communities ("subreddits"), restricted to comments and submissions made in 2024. We collect all users who post in both r/movies (the largest general movie discussion community) and at least one of five smaller specialized communities (r/horror, r/MovieSuggestions, r/Letterboxd, r/TrueFilm, and r/MovieDetails); we refer to the union of these five as the *alternative movie communities*. We treat each user's r/movies activity as the query profile and their activity in the alternative communities as the candidate profile; the candidate set contains all 9,781 users, so every query has a true match in the candidate set. Users share an average of 2.5 movie titles between the two splits (determined via fuzzy string matching of titles mentioned in posts), with 48% sharing only a single movie. We retain only users with at least one overlapping title, since no method restricted to movie-preference micro-data can match users with zero overlap.

5.2 Attack instantiation

Extract: structured movie reviews. We use LLMs to convert unstructured Reddit submissions—which may contain discussions, recommendations, or casual mentions—into structured movie reviews with movie titles, ratings, and dates. Ratings are typically estimated by the LLM based on sentiment. The resulting user profiles consist solely of movie reviews and contain no other micro-data.

Search: embedding similarity. As in Section 4, we encode the extracted review profiles into dense embeddings using Gemini embeddings and FAISS, and retrieve the top-100 nearest neighbors by cosine similarity on normalized vectors.

Reason: two-stage selection and verification. In contrast to Section 4, where we directly select from embedding neighbors, here we use a two-stage approach (Figure 5). First, we select from the top-K candidates using only reviews with matching movie titles between query and candidates. Then, we verify the top match using full profile text. This reduces cost—selection sees shorter inputs while verification runs only on the top candidate—and allows using a more capable model for verification. We use GPT-5.2 [27] with high reasoning effort.

Calibrate: LLM confidence scores. We use the verification-stage confidence as the calibration score for precision-recall curves; the selection-stage confidence is used only internally to pick the candidate passed to verification. The model outputs a numerical confidence score as a structured field in its function-call response, and we threshold it to trade off precision and recall. As a simpler alternative, we also evaluate using the gap between the top-2 embedding similarities as the confidence score—a large gap indicates the top candidate stands out clearly. As shown in Figure 4a, LLM confidence substantially outperforms this gap-based baseline.

Baseline. Both the LLM-based method and the classical baseline consume the same LLM-extracted (title, inferred rating, timestamp) tuples; they differ only in the matching procedure. As a classical baseline, we adapt the Netflix Prize attack: each user is represented as a vector of movie ratings (analogous to the Netflix Prize setting), and candidates are ranked by rarity-weighted similarity over shared movies, where popular movies are downweighted to emphasize distinctive preferences. See Appendix C for details.

5.3 Results

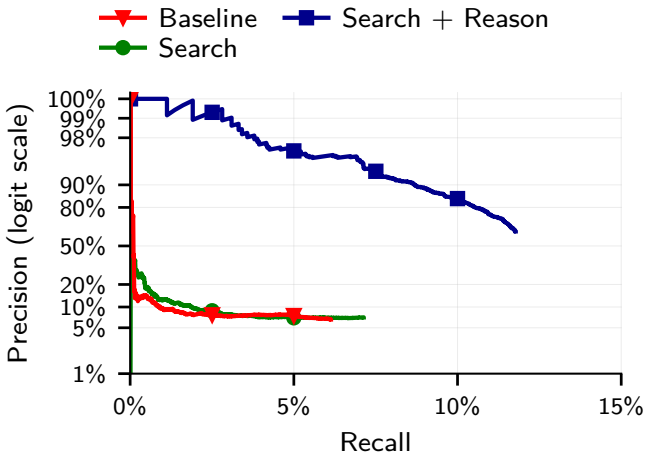
Higher reasoning effort substantially improves matching. We evaluate whether setting the reasoning effort to low or high affects the matching performance. High reasoning significantly outperforms low reasoning across all thresholds: at 90% precision, high reasoning achieves 8.5% recall compared to 5.2% for low reasoning (a relative improvement of 63%). At 99% precision, high reasoning achieves 2.8% recall versus 1.4% for low reasoning (a relative improvement of 100%). These differences are statistically significant ($p < 0.05$; see Table 5). Increased reasoning effort (test-time compute) substantially improves deanonymization success.

Users who share more content are substantially easier to identify. As shown in Figure 4b, recall increases sharply with the number of shared movies: at 90% precision, recall rises from 3.1% for users with one shared movie to 48.1% for users with ten or more.

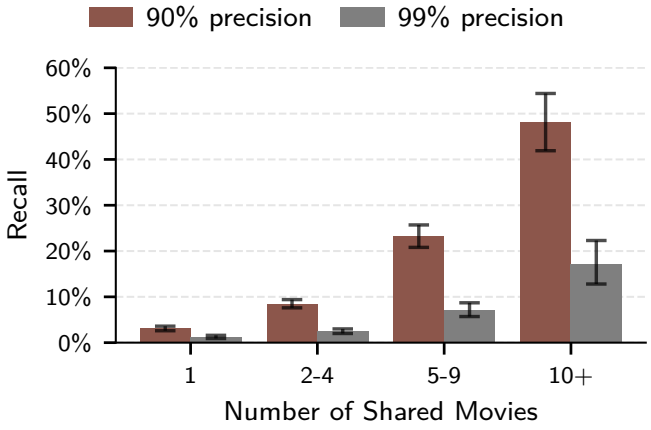
Sorting matches does not improve over LLM confidence scores. In Section 6, we introduce sorting matches with an LLM by perceived match likelihood as an alternative calibration method; we compare both approaches in Appendix F. We attempted to apply it to this setting on top of our two-stage selection and verification pipeline. However, sorting was unable to improve over the LLM confidence scores from the verification stage.

6 Scaling LLM deanonymization on temporally split profiles

As in the previous section, we split Reddit profiles to obtain large-scale ground truth. Here we split temporally rather than by community: we partition each user's comments into a before (query) and after (candidate) set, separated by a one-year gap. The one-year gap ensures that the matching task is non-trivial, as users' discussions of current events and short-term interests differ across splits. The resulting dataset allows us to study LLM-based deanonymization in depth and at scale. We also explore a different calibration approach based



(a) Precision-recall curves for Reddit movies matching.



(b) Recall by number of shared movies at different precision levels.

Figure 4: (a) Precision-recall curves comparing methods for Reddit movie matching. LLM reasoning outperforms both the embedding gap baseline and the Netflix Prize attack. (b) Recall at precision thresholds by number of shared movies, using GPT-5.2 high reasoning. Users who share more movies are substantially easier to identify. See Table 4 for detailed breakdown.

Stage 1: Selection from Top-k nearest neighbors

Query: *Neon Horizon* 7/10 "loved **Timothee** in the **rooftop fight**"
A: *Neon Horizon* 8/10 "great visuals"
B: *Neon Horizon* 6/10 "**Timothee** carried it, that **rooftop scene** was insane"
C: *Neon Horizon* 3/10 "**Timothee** was miscast, ruined the movie"

LLM Output: B (conf: 0.85). *"A lacks actor/scene details. C opposite sentiment on **Timothee**. B matches on **Timothee** + **rooftop**."*

Stage 2: Verification of selected match

Query: "loved **Timothee** in the **rooftop fight**, the choreography with the **neon signs** was so good"
B: "**Timothee** carried it, that **rooftop scene** with all the **neon lighting** was insane"

LLM Output: MATCH (conf: 0.94). *"Both mention **Timothee**, **rooftop scene**, and **neon** aesthetic—three specific details unlikely to co-occur by chance."*

Figure 5: Two-stage matching of movie reviews with simplified synthetic user profiles. The first LLM selects from top-K embedding similarity candidates; the second LLM verifies the selected match using full profile text. Bold text highlights profile details used by the LLM.

on pairwise comparisons rather than confidence scores, and we evaluate the attacks' robustness to difficulty parameters of our attack model.

6.1 Constructing the temporal-split dataset

We construct 10,000 query and candidate profiles as follows. We begin with 5,000 users, splitting each's comments into a query profile and a candidate profile. To increase difficulty, we add 5,000 candidate distractors: candidate profiles of users who appear only in the candidate pool, with no corresponding query. This collection of 5,000 query and 10,000 candidate profiles form the core matching task in Section 6.3, where every query has a match in the candidate set. To evaluate attack models where queries can be non-matchable (Section 6.4), we further add 5,000 query distractors: additional users who appear only in the query set, with no true match in the candidate pool. We construct two such datasets independently: a development set for tuning our attacks and a held-out test set for our final evaluation. All results presented in this section use the test set.

We apply several filters to ensure the matching task is meaningful and non-trivial. Our filters ensure that all 15,000 users are sufficiently active, but not too active. This yields split-profiles with sufficient micro-data while discarding bots. We also discard all comments within a one-year window around the split time of each user, so that the two split profiles do not share contemporaneous discussions of current events or short-term life circumstances. See Appendix G.1 for full details.

The temporal gap and difference in content between the split-profiles resembles what an attacker might face when linking an abandoned account to a newly created one or matching a user's main account to their alt-account. This requires identifying stable micro-data (e.g., user characteristics, interests, writing style, demographics) from hundreds of comments.

6.2 Attack instantiation

Extract: comment summaries. To extract micro-data features, we use LLMs to filter and summarize the comments of each split-profile. We first apply a two-stage relevance filter: a heuristic pre-filter removes empty and deleted comments, very short responses, and pure URLs. Then, we prompt Gemini 3

Flash to label each of the remaining comments as relevant or generic. We discard generic comments and feed the remaining relevant ones to Gemini 3 Pro [14]. The model generates a comma-separated list of the most important details, resulting in semi-structured micro-data. We discard both the query and candidate profile of users with zero comments after filtering (2 users) or if the LLM refuses to generate summaries due to inappropriate content (83 users). See Appendix G.2 for prompts and examples.

Search: cosine similarity over embedding. Similar to the previous two sections, we perform a nearest neighbor search (in terms of cosine similarity) between LLM embeddings of the extracted summaries. We generate the embeddings using OpenAI's `text-embedding-3-large` model.

The Extract and Search steps above yield a base attack: for each query, we return the candidate with the highest cosine similarity, using the similarity itself as the confidence for calibration. To justify the Reason and Calibrate steps, we investigate this base attack in more detail for the current setting. As detailed in Appendix G.4, the attack has two key issues. First, the embedding similarity successfully narrows the candidate pool but fails to reliably identify a single best guess. Second, embedding similarity is a poor confidence measure at high precision, such that high-similarity query-candidate pairs are not reliably correct matches. This motivates the design of the Reason and Calibrate steps.

Reason: LLM selection. As in the previous sections, we use LLMs to select the best match from the top-15 candidates (in terms of embedding similarity). We set $k = 15$ since we found that, on the training set, 80% of true matches fall within this range.[1] For each query, we give the 15 highest-scoring candidate summaries to Gemini 3 Pro, and ask it to select which best matches the query user's summary.

Calibrate: sorting matches via tournament. Since similarity scores are poorly calibrated confidence measures, we sort all proposed query-candidate matches from most to least plausible using pairwise LLM comparisons. Concretely, we implement a Swiss-system tournament over all selected query-candidate pairs (Algorithm 1), using the smaller GPT-5-mini [28] model for efficiency. In each round, pairs of matches are compared head-to-head: the LLM sees two query-candidate pairs and judges which is a more plausible match. After each round, we update Bradley-Terry ratings [3] based on the comparison outcomes. We run 15 rounds with approximately 2,500 comparisons per round, and output matches sorted by their final rating.

This sorting procedure avoids the quadratic cost of comparing all proposed query-candidate pairs, and we find it to be an effective confidence measure. However, sorting depends on interactions between query users; as such, it explicitly requires a large set of queries to be effective. This requirement makes sorting infeasible for attackers who aim to deanonymize a single user, but it is well-suited for large-scale attacks.

[1] The numbers are similar for the held-out test set; see Figure 8.

Algorithm 1 LLM-based confidence sorting

Require: Set of query-candidate pairs $\{(q_i, c_i)\}$ from the Reason step

Initialize Bradley-Terry [3] ratings r_i for all pairs
for round $= 1$ to N **do**
    Pair up matches by similar rating (Swiss-system matching)
    For each pair (q_i, c_i) vs. (q_j, c_j): LLM judges which is the more plausible match
    Update ratings r_i, r_j based on comparison outcome
end for
return pairs sorted by final rating r_i (descending)

Baseline: Netflix Prize attack on subreddit participation. Each user is represented as a binary vector indicating which subreddits the user posted a comment in. We then directly instantiate the Netflix Prize attack using this structured micro-data; see Appendix C for details.

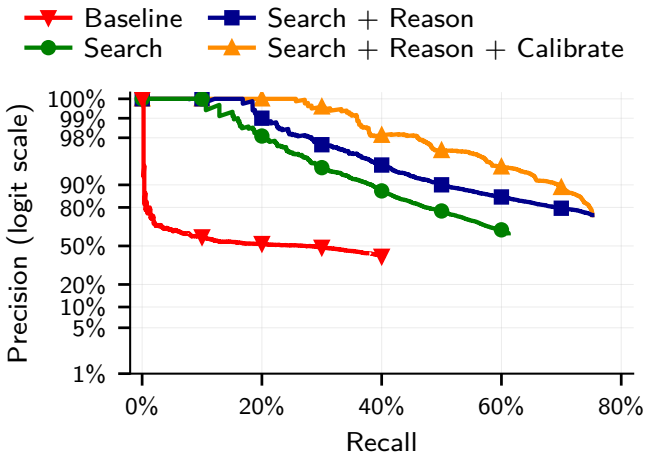
6.3 Results

We evaluate our LLM augmentations in the same attacker model as before, that is, with a fixed candidate pool that contains a ground truth match for every query user. Concretely, we run the attack for the 5,000 query profiles with a true match in the candidate set, and we fix the candidate pool to all 10,000 candidate profiles.

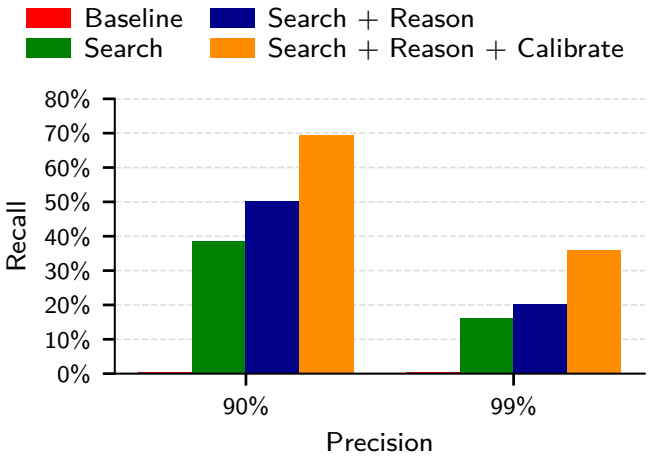
LLM-based extraction and search outperforms the classical baseline. As shown in Figure 6b, the classical attack, similar to the Netflix Prize attack, fails to achieve non-trivial recall at only 90% precision. In contrast, even our simplest attack achieves non-trivial recall at all precision levels.

LLMs are good at picking the correct match from a small set of candidates. Embedding similarity effectively narrows the candidate set: for about 80% of queries, the true match ranks among the top 15 candidates (see Figure 8a). Using an LLM to select the best candidate from the top 15, we recover many of these missed matches, increasing recall at high precision ("Search + Reason" in Figure 6a).

LLMs can prioritize more likely matches. The results in Figure 6 confirm our hypothesis that embedding similarity is a subpar confidence measure: Adding tournament-sorting significantly boosts recall across all precision values. In particular, our full attack ("Search + Reason + Confidence") reaches a recall closer to the best-possible value (80% imposed through the Reason step) at 90% precision, and it still deanonymizes a third of all users at 99% precision.



(a) Full Precision-Recall curves.



(b) Recall@Precision of the classical baseline (close to 0%) compared to LLM-based attacks.

Figure 6: **Classical attacks fail to deanonymize split Reddit profiles, while LLM-based attacks are highly effective.** We compare a classical baseline that mimics the Netflix Prize attack to LLM deanonymization. (a) The precision of classical attacks drops very fast, explaining its low recall. In contrast, the precision of LLM-based attacks decays more gracefully as the attacker makes more guesses. (b) The classical attack almost fails completely even at moderately low precision. In contrast, even the simplest LLM attack (Search) achieves non-trivial recall at low precision, and extending it with Reason and Calibrate steps doubles Recall@99% Precision .

6.4 Comparing difficulty parameters of our attack model

We now study the two key factors that determine the difficulty of our attack model (Section 3.1). As a reminder, a larger candidate pool makes it more challenging to find a correct match, and a lower a priori likelihood of there being a matching candidate requires more abstentions. Since these factors are typically unknown in practice, we investigate their full range in the following. This allows us to extrapolate how well our attacks can work in various real-world settings.

Setup. We rerun the baseline and our strongest LLM-based attack (Search + Reason + Calibrate) on candidate pools of various sizes. For each size, we sample subsets of the full 10k candidate profiles while ensuring that smaller pools are included in the larger ones (and the true match is always present). Since this procedure is random, we repeat it over five candidate set draws for the baseline and three draws for the (more expensive) LLM-based attack.

We further linearly extrapolate attack success to much larger candidate pools. Although our candidate pool of 10k users is reasonably large, platforms such as Reddit likely yield in the order of a million candidates, even after pre-filtering. Concretely, we fit a linear model to Recall@Precision as a function of $\log_{10}(K)$, where K is the size of the candidate set. To avoid overestimating attack success, we omit values for $K = 10$. This extrapolation paints a coarse picture of attack success in internet-scale settings.

For the second difficulty parameter, let $\pi \in [0,1]$ be the a priori probability that a query user has a matching candidate. Empirically, if M query-profiles are matchable and N profiles are not, $\pi = M/(M+N)$.

We can calculate Recall@Precision for all values of π post-hoc. First, notice that π only affects precision. Precision decreases through two types of error rates, depending on whether a query user has a true match in the candidate set:

- **False Match Rate (FMR)**: the fraction of *matchable* queries for which the attacker returns a wrong guess.
- **False Positive Identification Rate (FPIR)**: the fraction of *non-matchable* queries for which the attacker returns any guess (i.e., does not abstain).

As before, TPR (or recall) is the fraction of matchable queries for which the attacker returns the correct guess.

Precision is a function of those three rates and the fraction of matchable users π (see Appendix G.3 for the derivation):

$$\text{Precision}(\pi) = \frac{\pi \cdot \text{TPR}}{\pi \cdot \text{TPR} + \pi \cdot \text{FMR} + (1-\pi) \cdot \text{FPIR}}.$$

Crucially, the TPR and FMR only depend on the matchable queries, while the FPIR only depends on the non-matchable queries. This allows us to simulate different attack models: we first estimate the three rates and then use the reformulated precision as a plug-in estimator to calculate Recall@Precision for multiple values of π post-hoc. Concretely, we use the full

set of all 10k candidate profiles, but we run our attack on both the 5k query profiles with a true match in the candidate set and the additional 5k non-matchable query profiles.

We only evaluate the effects of the fraction of matchable users π for the Search and Search + Reason attacks. Since the tournament-based Calibration step correlates queries, we cannot calculate Recall@Precision post-hoc for our strongest attack. Moreover, the resulting metric has high variance for small π values. Thus, we would need to rerun the attack many times for accurate estimates, which is prohibitively expensive.

LLM-based attacks extrapolate to internet-scale datasets. As seen in Figure 7a, the recall of our LLM-base attack decreases roughly linearly in the scale of the dataset. Extrapolating to one million candidates, the LLM-based attack still yields about 35% recall at 90% precision. In contrast, the classical attack achieves a lower recall even for just a hundred candidates. We note that this is a coarse extrapolation, which should be read with large margins of error. Nevertheless, we conjecture that LLM deanonymization scales to internet-scale candidate pools with non-trivial success.

LLMs confidently deanonymize many users even if true matches are extremely rare. Figure 7b shows a surprising trend: LLM-based attacks consistently achieve a recall of at least 9% at 90% precision—even if the probability of a query having any match is only one in 10,000 ($\pi = 0.0001$). We hence conjecture that, even in settings where almost no users can be deanonymized, LLM-based attacks are reasonably likely to find a correct match for the few users that are identifiable. This demonstrates how LLM-based deanonymization still works in hard real-world settings.

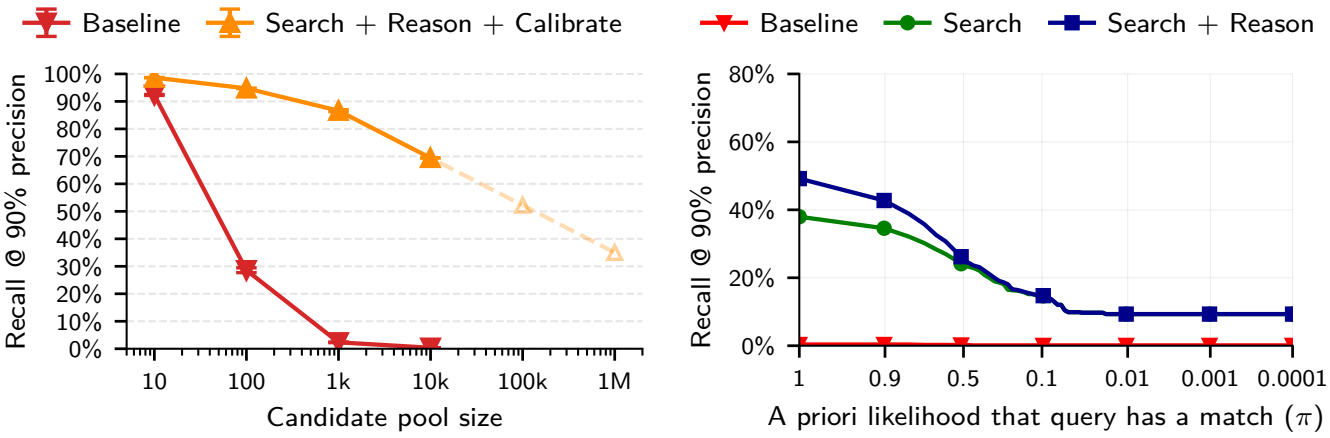
7 Related work

Deanonymization attacks predate LLMs. Narayanan and Shmatikov [24] demonstrated that movie ratings can uniquely identify individuals across platforms: by matching Netflix Prize data against public IMDb profiles, they deanonymized users at scale using statistical techniques on structured microdata. Their work established the threat model we build upon, showing that even sparse, seemingly innocuous data can be identifying when matched against auxiliary datasets. The same authors showed that social graph structure alone can deanonymize users by matching connection patterns across networks [25]. Wondracek et al. [36] exploited group membership information to deanonymize social network users, showing that the groups a user joins are often sufficient to uniquely identify them. More recently, Ederer et al. [12] deanonymized users of the Economics Job Market Rumors forum by exploiting weaknesses in its username generation scheme, recovering IP addresses for 66% of posts. These classical approaches required either structured features, exploitable technical vulnerabilities, or graph structure—none could operate on unstructured text at scale.

LLMs enable inference of personal attributes from unstructured text. The Extract step of our framework in Section 3.2 follows a line of work of LLM inferring demographic data from online profiles. Staab et al. [29] show that LLMs can infer personal attributes such as location, occupation, and income from text with high accuracy, demonstrating privacy risks beyond training data memorization. Du et al. [8] extend this line of work with AutoProfiler, a system of four specialized LLM agents that collaboratively extract sensitive attributes from pseudonymous platform activity. They report 85-92% accuracy in attribute extraction, demonstrating that automated profiling can be deployed at web scale. However, their evaluation of actual deanonymization (linking extracted attributes to real identities) relies on manual LinkedIn searches and reports k-anonymity metrics rather than the high-precision matching we focus on. Conversely, Staab et al. [30] show that LLMs can be used to anonymize text while preserving utility, suggesting a potential defensive application of the same capabilities. On the other hand, Xin et al. [38] and Krčo et al. [19] show that even after removing explicit identifiers, contextual details in text still leak sensitive attributes, challenging the assumption that surface-level sanitization provides meaningful privacy.

More broadly, the privacy community has begun shifting focus from training-data-centric concerns (memorization, extraction) toward deployment-time threats such as automated profiling and inference [9, 22]. Our work provides large-scale empirical evidence for these deployment-time risks in the deanonymization setting.

Recent work has begun exploring LLM-based deanonymization directly. Li [21] showed that agentic LLMs can deanonymize interview participants via web search, demonstrating that LLM agents make re-identification attacks low-effort: with a few natural-language prompts, off-the-shelf tools can search the web, cross-reference details, and propose matches. Their work on the Anthropic Interviewer dataset [2, 16] recovered 6 of 24 scientist identities by matching project descriptions to published papers, using task decomposition [18] to bypass safeguards. Apart from the mentioned AutoProfiler work on online platforms [8], several other recent works discuss re-identification in the context of redacted datasets. Nyffenegger et al. [26] evaluate LLM re-identification capabilities on court decisions, finding that despite high re-identification rates on Wikipedia, even the best LLMs struggled with anonymized legal documents. Xin et al. [38] propose a framework for evaluating re-identification risk in sanitized text datasets (medical records and chatbot conversations), finding that commercial PII removal leaves enough semantic signal for an adversary with auxiliary information to infer sensitive attributes. Our work extends this line of research by developing a systematic framework for LLM-based deanonymization and evaluating it across multiple platforms and attack settings.



(a) Effects of candidate set size

(b) Effects of the a priori likelihood that a query has a match (π)

Figure 7: **LLM deanonymization scales to more challenging attack models, while classical methods fail.** (a) The strongest LLM-based attack (Search + Reason + Calibrate) might remain effective for very large candidate pools, while the classical baseline fails to deanonymize any users given much smaller candidate pools. We log-linearly extrapolate the recall from sizes 100, 1k, and 10k. (b) LLM-based attacks gracefully degrade as matchable users become more rare, converging to around 9% recall even if only one in 10k queries has a possible match. In contrast, the classical baseline fails to deanonymize users even if all of them are matchable ($\pi = 1$). See Appendix G.5 for more precision levels.

Stylometry identifies users through writing style rather than semantic content. Authorship attribution uses features such as function word distributions and syntactic structures [31]; recent work applies LLM embeddings to this task [35]. Determining whether two texts were written by the same person has been studied extensively through the PAN shared task series [32], which evaluates pairwise verification on corpora of around 100 authors using stylometric features. Authorship attribution approaches explicitly treat topical similarity as a confound to be controlled for, whereas our method exploits semantic content as the primary signal. Our temporal-split Reddit setting (Section 6) is related to authorship linking, but operates at larger scale (10,000 users) and relies on *what* users say rather than just *how* they say it. Combining stylometric and semantic approaches is an interesting direction for future work.

More broadly, deanonymization is one of many ways LLMs empower both criminals and state actors. Carlini et al. [4] argue that LLMs alter the economics of cyberattacks by enabling adversaries to launch tailored attacks on a user-by-user basis, fundamentally changing the cost-benefit calculus for attackers. Heiding et al. [17] demonstrate that LLM agents can autonomously crawl public information to construct profiles that were comprehensive for 88% of targets, using them to generate spear phishing emails with click-through rates on par with human experts.

8 Discussion

The future of online privacy. Deanonymization is one instance of LLMs acting as an "information microscope" that makes previously manual and expensive attacks scalable [15]. Our paper shows that LLMs democratize deanonymization. Echoing concerns raised by prior work on LLM-based attribute inference and semantic privacy leaks [22, 29, 38], we argue that the asymmetry between attack cost and defense cost may force a fundamental reassessment of what can be considered private online. Our large-scale experiments provide quantitative evidence for these concerns in the deanonymization setting.

So what do our findings mean for the future of privacy? Governments could link pseudonymous accounts to real identities for surveillance of dissidents, journalists, or activists. Corporations could connect seemingly anonymous forum posts to customer profiles for hyper-targeted advertising. Attackers could build sophisticated profiles of targets at scale to launch highly personalized social engineering scams. Hostile groups could identify important employees and decision makers and build online rapport with them to eventually leverage in various forms. Users, platforms, and policymakers must recognize that the privacy assumptions underlying much of today's internet no longer hold.

Possible mitigations. Enforcing a rate limit for API access to user data, detecting automated scraping, and restricting bulk data exports may reduce the severity of these attacks. LLM providers could also monitor the use of their models to detect misuse such as deanonymization attempts [1, 33]. Improved

safety guardrails that make models refuse deanonymization requests might also provide some benefit, although our methods use LLMs in many ways that resemble benign usage (summarization of profiles, computing embeddings, etc.).

Classical anonymization frameworks such as k-anonymity [34] and differential privacy [10, 11] were designed for structured databases with explicit identifiers and assume attackers use direct matching or statistical queries. These frameworks do not account for the types of attacks we demonstrate; data releases should consider such threats when evaluating privacy risks. LLM-based text anonymization [30] offers a more targeted defense for unstructured text, though even these methods leave residual semantic signals [38].

Generalizability of our results. Our evaluation relies on ground truth datasets that may overestimate real-world success rates. Users who publicly link their accounts or share identifying information may still share more information than they would for truly pseudonymous accounts, even considering our anonymization step. However, measuring deanonymization performance requires ground truth, and we cannot verify matches for users who have not revealed their identities. While there are synthetic benchmarks for attribute inference [39], it is not clear how to construct realistic synthetic datasets for deanonymization. Profile splitting has different limitations: users might behave differently across truly separate platforms than across different communities or across time on the same site. However, our methods work across a broad range of experimental setups, which suggests that they do generalize beyond any single evaluation setting.

Platforms should therefore assume that pseudonymous users can be linked across accounts and to real identities at scale. This should influence the decisions they make on data access policies. Users should similarly not assume that posting under a pseudonym provides meaningful protection.

Are our evaluations contaminated? One might wonder whether LLMs succeed at deanonymization because they memorized Reddit or Hacker News data during training. The fact that increasing reasoning effort substantially improves performance (Sections 4 and 5) provides some evidence that reasoning plays a significant role and memorization alone cannot explain the results. The training data for LLMs is typically not openly revealed, making it challenging to isolate these factors. We suspect that Hacker News and Reddit are part of most training corpora, but LinkedIn profiles are not. More broadly, even if memorization plays a role, this does not diminish the privacy implications: many social media platforms are included in LLM training corpora, so the deanonymization threat would only be reduced for platforms excluded from training data.

9 Conclusion

We have demonstrated that LLMs enable deanonymization of pseudonymous online accounts at scale, outperforming classical methods. In many cases, LLMs enable us to perform attacks that would not have previously been possible, due to the lack of structured data or features.

These attacks require only publicly available models and standard APIs. Our pipeline uses only publicly available embedding models, standard LLM APIs, and LLM-agent scaffolding, placing them within reach of moderately resourced adversaries.

Pseudonymity does not provide meaningful protection online. Users who post under persistent usernames should assume that adversaries can link their accounts to real identities or to each other, and that the probability rises with each piece of micro-data they post.

Preventing such attacks appears challenging. Not revealing any data on online platforms is difficult, as the data we use is the very data that makes online communities worthwhile. Although LLM providers could aim to detect and block attempts to misuse their models for deanonymization (as they do, for instance, for cyberattacks), we are pessimistic that this is possible as our deanonymization framework splits an attack into a combination of seemingly benign summarization, search and ranking tasks.

Recent advances in LLM capabilities have made it clear that there is an urgent need to rethink various aspects of computer security in the wake of LLM-driven offensive cyber capabilities. Our work shows that the same is likely true for privacy as well.

Ethical Considerations

Stakeholder analysis. The primary stakeholders are social media users whose privacy could be compromised by deanonymization attacks. General social media users who post under pseudonyms expecting privacy, vulnerable populations who depend on anonymity (activists, abuse survivors, whistleblowers), and the specific users whose data we used in experiments (Reddit users, HN users, etc.). Secondary stakeholders include platforms, researchers, potential malicious actors, and society broadly.

Potential harms. The primary potential harm is that publishing this research could inspire malicious deanonymization attacks. Concrete harms include stalking and harassment, doxxing of activists or vulnerable individuals, corporate surveillance and targeted manipulation, government surveillance and suppression of dissent, and chilling effects on free speech if people fear being identified. However, these capabilities already exist in deployed LLMs; we are not introducing novel attack vectors but documenting existing risks.

Potential benefits. The primary benefit is raising awareness of privacy risks that already exist due to widely available LLM capabilities. Users can make informed decisions about what they share online and add better privacy measures. Platforms can develop better privacy protections and reconsider making data publicly available, such as for LLM training. Policymakers can consider appropriate regulations, and LLM providers can consider adding additional safety guardrails that prevent large scale misuse. The security community can develop defenses and metrics, similar to concepts such as k-anonymity. Before a dataset is irreversibly publicly released, researchers could study whether the information could be used by LLM agents to identify individuals.

Mitigations. We designed our experimental methodology to avoid directly harming individuals. Most experiments do not deanonymize individuals, and we instead used synthetically constructed datasets (profile splitting, LLM-anonymized data). In the case of the Anthropic interview dataset, we note that a previous paper had already performed a similar attack [2]. We do not reveal any names or identities in this paper. We do not release our matching pipeline code or processed datasets.

Decision to publish. We believe that the benefits of publication outweigh the marginal risks because these capabilities are already widely available. Any moderately sophisticated actor can already do what we do using readily available LLMs and embedding models. With future LLMs, without mitigations, this attack will be within the means of basically all adversarial actors. By documenting the threat while it is nontrivial to execute, we enable proactive responses. The privacy community, LLM providers, online platforms and users need to know about these risks, so not publishing would leave users unaware and unprotected.

Open Science

Balancing the requirements of reproducibility while avoiding undue harm is challenging for works on deanonymization. Since defenses typically do not exist once the data is available, prior work has often refrained from releasing code to reproduce their attacks (e.g., [5, 21, 24]).

Although many of our experiments are conducted with publicly available data that we synthetically anonymized for experimental purposes, we still believe that releasing this data and associated code would do more harm than is warranted by strict reproducibility concerns. Indeed, the data consist of real user profiles that, while publicly available at the time of writing, should not be made available as research artifacts. Releasing code to directly run our attacks would needlessly make it trivial for someone to attempt a similar attack on other truly anonymous online profiles.

Ethics

This study was approved by ETH Zurich's Ethics Review Board (ERB).

Author Contributions and Acknowledgments

This research was directed by ETH Zurich, which conducted the primary experiments. Nicholas Carlini advised on the research. We thank Robin Staab and Mark Verö for extensive feedback.

References

[1] Anthropic. Disrupting the first reported ai-orchestrated cyber espionage campaign. Technical report, Technical report, 11 2025.

[2] Anthropic. Anthropic Interviewer dataset, December 2025. URL https://huggingface.co/datasets/Anthropic/AnthropicInterviewer. Archived at https://web.archive.org/web/20251207050016/https://huggingface.co/datasets/Anthropic/AnthropicInterviewer.

[3] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[4] Nicholas Carlini, Milad Nasr, Edoardo Debenedetti, Barry Wang, Christopher A. Choquette-Choo, Daphne Ippolito, Florian Tramèr, and Matthew Jagielski. LLMs unlock new paths to monetizing exploits. *arXiv preprint arXiv:2505.11449*, 2025. URL https://arxiv.org/abs/2505.11449.

[5] Aloni Cohen. Attacks on deidentification's defenses. In *31st USENIX security symposium (USENIX Security 22)*, pages 1469–1486, 2022.

[6] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3(1):1376, 2013.

[7] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The FAISS library. *IEEE Transactions on Big Data*, 2025.

[8] Yuntao Du, Zitao Li, Bolin Ding, Yaliang Li, Hanshen Xiao, Jingren Zhou, and Ninghui Li. Automated profile inference with language model agents. *arXiv preprint arXiv:2505.12402*, 2025.

[9] Yuntao Du et al. Beyond data privacy: New privacy risks for large language models. *arXiv preprint arXiv:2509.14278*, 2025.

[10] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 4052, pages 1–12, 2006.

[11] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[12] Florian Ederer, Paul Goldsmith-Pinkham, and Kyle Jensen. Anonymity and identity online. *arXiv preprint arXiv:2409.15948*, 2024.

[13] Jennifer Lynnae Garcia. The evidentiary trail down Silk Road. Master's thesis, Boston University – Metropolitan College, 2017. URL https://www.researchgate.net/publication/319164300_The_Evidentiary_Trail_Down_Silk_Road.

[14] Google DeepMind. Gemini 3 pro model card. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf, 2025. Model card updated December 2025. Accessed: 2026-02-18.

[15] Samuel Hammond. AI and Leviathan: Part I, 2025. URL https://www.secondbest.ca/p/ai-and-leviathan-part-i.

[16] Karina Handa, Max Stern, Sandy Huang, Jessica Hong, Esin Durmus, Miles McCain, Grace Yun, Alex Alt, Tiffany Millar, Alex Tamkin, Julia Leibrock, Stuart Ritchie, and Deep Ganguli. Introducing Anthropic Interviewer: What 1,250 professionals told us about working with AI, 2025. URL https://anthropic.com/research/anthropic-interviewer. Archived at https://web.archive.org/web/20251204184855/https://www.anthropic.com/research/anthropic-interviewer.

[17] Fred Heiding, Simon Lermen, Andrew Kao, Claudio Mayrink Verdun, Bruce Schneier, and Arun Vishwanath. Evaluating large language models' ability to automate spear phishing. *Expert Systems with Applications*, 314:131546, 2026. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2026.131546. URL https://www.sciencedirect.com/science/article/pii/S0957417426004598.

[18] Erik Jones, Anca Dragan, and Jacob Steinhardt. Adversaries can misuse combinations of safe models. *arXiv preprint arXiv:2406.14595*, 2024.

[19] Nataša Krčo, Zexi Yao, Matthieu Meeus, and Yves-Alexandre de Montjoye. Rat-bench: A comprehensive benchmark for text anonymization. *arXiv preprint arXiv:2602.12806*, 2026.

[20] Jinhyuk Lee et al. Gemini embedding: Generalizable embeddings from Gemini. *arXiv preprint arXiv:2503.07891*, 2025.

[21] Tianshi Li. Agentic LLMs as powerful deanonymizers: Re-identification of participants in the Anthropic Interviewer dataset. *arXiv preprint arXiv:2601.05918*, 2026.

[22] Niloofar Mireshghallah and Tianshi Li. Position: Privacy is not just memorization! *arXiv preprint arXiv:2510.01645*, 2025.

[23] John Morris, Justin Chiu, Ramin Zabih, and Alexander M Rush. Unsupervised text deidentification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4777–4788, 2022.

[24] Arvind Narayanan and Vitaly Shmatikov. Robust deanonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125, 2008. doi: 10.1109/SP.2008.33.

[25] Arvind Narayanan and Vitaly Shmatikov. Deanonymizing Social Networks. In *2009 30th IEEE Symposium on Security and Privacy*, pages 173–187. IEEE, 2009. doi: 10.1109/SP.2009.22.

[26] Alex Nyffenegger, Matthias Stürmer, and Joel Niklaus. Anonymity at risk? Assessing re-identification capabilities of large language models in court decisions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2433–2462, Mexico City, Mexico, jun 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.157. URL https://aclanthology.org/2024.findings-naacl.157/.

[27] OpenAI. GPT-5.2 system card. Technical report, OpenAI, December 2025. URL https://cdn.openai.com/pdf/3a4153c8-c748-4b71-8e31-aecbde944f8d/oai_5_2_system-card.pdf. Technical report on GPT-5.2 model capabilities and safety evaluations.

[28] OpenAI. Openai gpt-5 system card, 2025. URL https://arxiv.org/abs/2601.03267.

[29] Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Beyond memorization: Violating privacy via inference with large language models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ySXVKYNzad.

[30] Robin Staab, Mark Vero, Mislav Balunovic, and Martin T. Vechev. Large language models are advanced anonymizers. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=82p8VHRsaK.

[31] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[32] Efstathios Stamatatos, Krzysztof Kredens, Piotr Pezik, Annina Heini, and Janek Janićki. Overview of the authorship verification task at PAN 2023. In *Working Notes of CLEF 2023*, volume 3497 of *CEUR Workshop Proceedings*, 2023.

[33] Theodore Sumers, Raj Agarwal, Nathan Bailey, Tim Belonax, Brian Clarke, Jasmine Deng, Evan Frondorf, Kyla Guru, Keegan Hankes, Jacob Klein, Lynx Lean, Kevin Lin, Linda Petrini, Madeleine Tucker, Ethan Perez, Mrinank Sharma, and Nikhil Saxena. Monitoring computer use via hierarchical summarization, 2025. URL https://alignment.anthropic.com/2025/summarization-for-monitoring.

[34] Latanya Sweeney. k-Anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[35] Jacob Tyo, Bhuwan Dhingra, and Zachary C Lipton. On the state of the art in authorship attribution and authorship verification. *arXiv preprint arXiv:2209.06869*, 2022.

[36] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to de-anonymize social network users. In *2010 IEEE Symposium on Security and Privacy*, pages 223–238. IEEE, 2010. doi: 10.1109/SP.2010.21.

[37] xAI. Grok 4.1 fast and agent tools api. https://x.ai/news/grok-4-1-fast, November 2025. Agentic tool-calling model with 2M token context window.

[38] Rui Xin, Niloofar Mireshghallah, Shuyue Stella Li, Michael Duan, Hyunwoo Kim, Yejin Choi, Yulia Tsvetkov, Sewoong Oh, and Pang Wei Koh. A false sense of privacy: Evaluating textual data sanitization beyond surface-level privacy leakage. *arXiv preprint arXiv:2504.21035*, 2025.

[39] Hanna Yukhymenko, Robin Staab, Mark Vero, and Martin Vechev. A synthetic dataset for personal attribute inference. *Advances in Neural Information Processing Systems*, 37:120735–120779, 2024.

A Profile anonymization procedure

To prevent trivial deanonymization via unique identifiers while preserving semantic content relevant to our matching task, we apply anonymization rules based on whether searching the identifier online would directly reveal the profile owner.

Category	Example	Search Result	Action
Personal URL	knuth.stanford.edu	Finds Donald Knuth	Remove
Blog URL	karpathy.bearblog.dev	Finds Andrej Karpathy	Remove
Social handle	u/spez	Finds Steve Huffman	Remove
GitHub repo	flask	Finds Armin Ronacher	Remove
GitHub handle	torvalds	Finds Linus Torvalds	Remove
Named bootcamp	"F*****"	Finds creator	Generalize
Founded project	"C****"	Finds founder	Generalize
Mentioned colleague	Yann LeCun	Finds Meta	Keep
Local business	Cafe Carpe Diem	Finds the city	Keep
Institution	UCLA	Too broad	Keep
Demographics	Male, 40s	Too broad	Keep
Interests	plays chess	Too broad	Keep
Tech stack	uses LaTeX	Too broad	Keep

Table 2: Anonymization rules for profile data, closely matching our actual implementation. We remove direct identifiers, generalize unique project names, and keep information that does not uniquely identify the profile owner.

Remove entirely. We delete lines containing personal website/blog URLs, social media handles, and GitHub repository names or handles—any identifier where searching online directly reveals the profile owner.

Generalize. We replace unique project names with generic descriptions (e.g., "Founder of <specific programming community>" becomes "Founded a programming community").

Keep as-is. We retain colleague names (searching finds them, not the profile owner), local business names (searching finds the business, not its patrons), institution names, locations, demographics, interests, and technical skills.

B Agentic deanonymization experiments on Reddit

We report full results for the two Reddit experiments summarized in Section 2.

Reddit academics. We curate 25 identifiable Reddit users posting on academic subreddits whose usernames reveal their real names. We manually verified each ground-truth identity, excluding ambiguous cases. Then we anonymize their profiles to remove any direct identifiers, such as names or paper titles, but preserve any research interests or institutions they mention on their social media profile. We run our de-anonymizing agent on this dataset, and it identifies 13 of 25 (52%) with 72% precision (5 incorrect, 7 abstentions).

Reddit career discussions. We collect 36 users who had posted their own LinkedIn URL for resume review on a software engineering career subreddit. We manually check each ground-truth identity, and exclude ambiguous cases. After anonymizing their Reddit profiles, the agent correctly identifies 9 of 36 (25%) at 90% precision (1 incorrect, 26 absten-

tions). The lower recall likely reflects that career discussion posts are less uniquely identifying: when we remove direct identifiers, many user's comments discuss generic software engineering topics without mentioning employers, projects, or credentials that would distinguish them from other users.

C Baseline attack details

We adapt the approach of Narayanan and Shmatikov [24] to our datasets.

LinkedIn to Hacker News. For professional profile matching, each user is represented as a binary vector $\mathbf{x} \in \{0,1\}^d$ where $d = 332$ is the vocabulary size, with each dimension corresponding to an attribute (e.g., `skill_python`, `company_google`). Given a query user with feature set F_a and a candidate with feature set F_b, we compute weighted Jaccard similarity:

$$\text{sim}(a,b) = \frac{\sum_{f \in F_a \cap F_b} w_f}{\sum_{f \in F_a \cup F_b} w_f}$$

where the rarity weight $w_f = 1/\log(1+c_f)$ downweights common attributes, and c_f is the frequency of feature f across all users. For each query user, we rank all candidates by descending similarity. For LinkedIn profiles, we parse the binary vector $\mathbf{x}$ directly from the structured data we collected; for HN profiles, we use an LLM with function calling to extract the same attributes from unstructured text.

Reddit movies. For movie review matching, we use a weighted similarity that accounts for rating agreement and temporal proximity:

$$\text{sim}(a,b) = \frac{\sum_{m \in M_a \cap M_b} w_m \cdot s_m}{\sum_{m \in M_a \cup M_b} w_m}$$

where M_a, M_b are the sets of movies rated by users a and b. The per-movie similarity combines a rating kernel and a temporal kernel:

$$s_m = k_r^{\beta} \cdot k_t^{1-\beta}$$

$$k_r = \exp\left(-\frac{|r_a^m - r_b^m|}{\sigma_r}\right), \quad k_t = \exp\left(-\frac{|t_a^m - t_b^m|}{\sigma_t}\right)$$

where r_a^m is user a's rating for movie m and t_a^m is the timestamp (in days). The rarity weight $w_m = 1/\log(1+c_m)$ downweights popular movies, where c_m is the total rating count for movie m. We use $\sigma_r = 1.0$, $\sigma_t = 40.0$ days, and $\beta = 0.5$.

Unlike Narayanan and Shmatikov [24] who used a binary kernel (outputting 1 if reviews fall within a time threshold and have similar ratings, 0 otherwise), we use a smooth kernel because ratings and dates are heavily inferred by the LLM. Users rarely give explicit numerical ratings in Reddit comments, so the model must estimate scores from sentiment. Similarly, while we observe comment timestamps, a user's discussion of a single movie may span multiple comments over weeks or months, requiring the model to assign an approximate review date. Due to this inherent noise in LLM-extracted features, we use a more forgiving kernel that still rewards temporal and rating proximity.

Reddit temporal. For temporal profile matching, each user is represented as a binary vector $\mathbf{x} \in \{0,1\}^d$ where d is the number of subreddits and $x_s = 1$ if the user posted in subreddit s. Given a query user with subreddit set S_q and a candidate with subreddit set S_c, we compute a weighted intersection score:

$$\text{sim}(q,c) = \sum_{s \in S_q \cap S_c} w_s$$

where the rarity weight $w_s = 1/\log|U_s|$ and $|U_s|$ is the number of users who posted in subreddit s. This follows the original Netflix Prize attack formulation: rare subreddits (with fewer users) provide stronger evidence of a match. Unlike normalized metrics such as Jaccard or cosine similarity, this score is unbounded and favors users who post in many rare subreddits.

D LinkedIn to HN matching: additional details

D.1 Scaling analysis

We evaluate how recall at 90% precision scales with candidate pool size by subsampling from our full 89k HN user pool. Table 3 shows recall for Search and Reason (high) at different pool sizes.

Table 3: Recall @ 90% precision by candidate pool size. Starred rows show log-linear extrapolated values.

Candidate pool	Reason (high)	Search (embeddings)
10	90.0%	80.0%
100	82.0%	74.0%
1k	68.3%	47.6%
10k	63.2%	40.3%
30k	59.0%	35.1%
89k	55.2%	26.6%
1M*	*45.1%*	*12.1%*
10M*	*36.2%*	*0.0%*
100M*	*27.4%*	*0.0%*

Both methods degrade log-linearly with pool size. We fit a log-linear model:

$$\text{Recall}(\%) = a \cdot \log_{10}(N) + b$$

where N is the candidate pool size, fitted via ordinary least squares on the measured data points. For Reason: $a = -8.88$, $b = 98.35$; for Search: $a = -13.94$, $b = 95.73$. Extrapolated values for 1M, 10M, and 100M candidates are shown in italics in Table 3. Reason degrades much more gracefully: at 100M

candidates it would still retain an estimated 27% recall at 90% precision, while Search falls to zero.

E Reddit movies: additional details

E.1 Recall by shared movies

Table 4 provides a detailed breakdown of recall at different precision thresholds, stratified by the number of movies a user discussed in both r/movies and the alternative movie communities. Users with more shared movies are substantially easier to identify.

Table 4: Recall at precision thresholds by number of shared movies (total queries: 9,781). 95% Wilson CIs shown.

#Shared	n	90% Prec	99% Prec
1	4,729	3.1% (2.6–3.6)	1.2% (0.9–1.6)
2–4	3,693	8.4% (7.6–9.4)	2.5% (2.0–3.0)
5–9	1,118	23.2% (20.8–25.7)	7.1% (5.7–8.7)
10+	241	48.1% (41.9–54.4)	17.0% (12.8–22.3)
Overall	9,781	8.5% (8.0–9.1)	2.8% (2.4–3.1)

E.2 Reasoning effort comparison

Table 5 compares recall at different precision thresholds when using low versus high reasoning effort on GPT-5.2. High reasoning effort consistently outperforms low reasoning, roughly doubling recall at 99% precision (2.8% vs. 1.4%).

Table 5: Recall at different precision thresholds for Reddit movies matching (9,781 queries) by reasoning effort. Reason uses GPT-5.2. 95% Wilson CIs shown.

Method	90% Prec	98% Prec	99% Prec
Reason (low)	5.2% (4.8–5.7)	2.0% (1.7–2.3)	1.4% (1.1–1.6)
Reason (high)	**8.5%** (8.0–9.1)	**3.8%** (3.4–4.2)	**2.8%** (2.4–3.1)

F Comparing calibration methods: confidence scores vs. sorting

There are two ways to use LLMs in the Calibrate step of our framework: (1) ask the LLM to output confidence scores directly alongside its match decision, or (2) sort all proposed matches by running pairwise comparisons. We use the first approach in Sections 4 and 5, where the LLM outputs a structured confidence score during the Reason step, and we threshold this score to trace precision-recall curves. We use the second approach—Swiss-system sorting with Bradley-Terry ratings—in Section 6, where it yields large gains over embedding similarity as a confidence measure.

These two approaches differ in what information they exploit. Direct confidence scoring evaluates each query-candidate pair independently: the LLM sees a single proposed match and estimates how plausible it is. Pairwise sorting compares proposed matches against each other: the LLM sees two query-candidate pairs side by side and judges which is more plausible. Sorting thus has access to cross-query information that direct scoring does not, but it also requires a large batch of queries and incurs additional LLM calls (approximately $O(n \log n)$ comparisons for n proposed matches).

Movie matching experiment. We attempted to apply pairwise sorting to the Reddit movie matching task from Section 5. On top of all matches confirmed by the high-reasoning GPT-5.2 verification stage, we ran 15 rounds of pairwise comparisons using `gpt-5-mini-2025-08-07`.

Results were worse than using the verification-stage confidence scores directly. At 99% precision, sorting achieved only 0.7% recall compared to 3.1% for confidence ranking. At 90% precision, sorting achieved 3.6% recall versus 8.7% for confidence. At 80% precision, sorting achieved 7.0% recall versus 10.5% for confidence.

We note that the verification stage uses a much more powerful model (GPT-5.2 with high reasoning effort) than sorting (GPT-5-mini). Additionally, sorting's $O(n \log n)$ LLM calls make it less suitable for expensive models or larger dataset sizes. These results suggest that when the Reason step already uses a strong model with high reasoning effort, sorting provides limited additional benefit.

G Linking temporally split Reddit profiles: additional details

G.1 Data processing pipeline

We start from publicly available Reddit comment dumps from Academic Torrents, then apply a multi-stage filtering pipeline to select suitable authors and comments. We first scan all comment dumps to aggregate per-author statistics (total comment count, date range of activity), then apply the following filters:

1. Activity span: at least 1,095 days (3 years) between first and last comment.
2. Minimum volume: at least 200 total comments.
3. Rate limit: at most 24 comments per day on average. We find that this simple heuristic is highly effective in excluding automated accounts and bots.

4. Bot exclusion: usernames ending in `bot`, `gpt`, or `mods` are removed, along with a manually curated bot list.

This yields an initial filtered author set. We then extract all comments for these authors and build an index of comment timestamps per author.

We then split each author's comment history into two disjoint profiles separated by a temporal gap. For each author, we find the split time t^* with gap window $g = 365$ days which maximizes the number of comments in the smaller of the resulting halves, subject to a minimum of 100 comments on each side. Comments within the gap, i.e., in the interval $[t^* - g/2,\ t^* + g/2]$, are discarded. This produces two temporally separated split-profiles per author:

- Query half (before): all comments with timestamp $< t^* - g/2$.
- Candidate half (after): all comments with timestamp $> t^* + g/2$.

The gap ensures that the matching task is more difficult and more realistic. We exclude authors who do not satisfy the 100-comment minimum on both sides, and shuffle the remaining authors with a fixed random seed for reproducible sampling. We select 15,000 authors for our primary dataset.

Comments are then deduplicated and filtered for relevance before feature extraction. During data loading, duplicate comments (same author and timestamp) are dropped. We apply a two-stage relevance filter when running the LLM-based embedding method:

1. **Heuristic pre-filter**: removes deleted/removed comments, ultra-short comments ($\leq$3 characters), and pure URL comments.
2. **LLM-based semantic filter**: batches of comments are sent to a smaller LLM (Gemini 3 Flash) that identifies generic or non-personal content (e.g., one-word replies in large default subreddits). Each comment is labeled as relevant or irrelevant. Only relevant comments are retained for summarization.

G.2 Prompts

Filtering. The following prompt is used with Gemini 3 Flash to pre-filter generic and uninformative comments.

Filtering Prompt

```
Your task is to identify Reddit comments that
should be filtered out. All comments below are
from the SAME author, listed in CHRONOLOGICAL
ORDER (oldest first). A comment should be
FILTERED if:
1. It reveals nothing about the author, OR
2. It reveals information already shown in an
earlier comment (redundant)

CRITICAL: Check the subreddit FIRST

Before looking at comment content, check if
the subreddit itself reveals information.
If YES, KEEP the FIRST comment from that
subreddit, but FILTER subsequent comments from
the same revealing subreddit (redundant - we
already know they participate there).

REVEALING SUBREDDITS (KEEP first occurrence,
FILTER subsequent):
- Health/medical: r/diabetes, r/celiac, r/adhd,
r/anxiety, r/depression, etc.
- Cities/regions: r/nyc, r/seattle, r/Portland,
r/austin, r/london, r/toronto, etc.
- Professions: r/teachers, r/nursing,
r/medicine, r/law, r/sysadmin, etc.
- Religion: r/atheism, r/christianity, r/islam,
r/judaism, etc.
- Identity: r/actuallesbians, r/gaybros,
r/lgbt, r/transgender, r/childfree, etc.
- Sports teams: r/bostonceltics, r/lakers,
r/49ers, r/cowboys, etc.
- Niche hobbies: r/woodworking, r/knitting,
r/homebrewing, r/beekeeping, etc.

GENERAL-INTEREST SUBREDDITS (subreddit alone
reveals nothing):
- r/AskReddit, r/funny, r/pics, r/videos,
r/gaming, r/movies, r/music, r/aww, etc.

Then check comment content for NEW information

FILTER if:
- Generic reactions: "lol", "this", "same",
"agreed", "nice", "wow", "thanks"
- Information already revealed in an earlier
comment (e.g., second mention of "my wife")

KEEP if it reveals NEW information about:
- Demographics (age, gender, location,
occupation)
- Personal circumstances ("my wife", "my kids",
"when I was in college")
- Opinions or values (political, religious,
lifestyle stances)
- Expertise or interests shown through
knowledge

EXAMPLE (same author):
[0] [r/Portland] Just moved here last year
[1] [r/diabetes] My A1C is finally under
control
[2] [r/Portland] The food carts are amazing
[3] [r/gaming] I'm 34 and still play regularly
[4] [r/diabetes] Same here, diet really helped
[5] [r/pics] lol
[6] [r/AskReddit] My wife and I love hiking
[7] [r/videos] My wife showed me this
```

```
Output: 2,4,5,7

Explanation:
- [0] KEEP: r/Portland reveals location (first
occurrence)
- [1] KEEP: r/diabetes reveals health
condition (first occurrence)
- [2] FILTER: r/Portland redundant (already
know they're in Portland from [0])
- [3] KEEP: reveals age_30s (new info)
- [4] FILTER: r/diabetes redundant (already
know from [1])
- [5] FILTER: generic "lol" in
general-interest sub
- [6] KEEP: reveals married (new info)
- [7] FILTER: "my wife" redundant (already
know from [6])

Output ONLY the indices of comments to FILTER.
Comma-separated, no spaces (e.g. 0,3,7). If
none should be filtered, output "none".
COMMENTS: comments
```

Summarization. The following prompt is used with Gemini 3 Pro to summarize split profiles based on their comments.

Summarization Prompt

```
Below are comments from a single Reddit user.
Output a comma-separated list of DISTINCTIVE
traits that would help identify this specific
person among thousands of random Reddit users.

CRITICAL: Focus on traits that are BOTH
distinctive AND persistent over time. If
you saw this user's comments from a different
year, these traits should still be true.

SKIP these (too common to be useful):
- uses_profanity, analytical_tone,
sarcastic_tone, informal_tone, verbose_style
- Generic "gaming_interest" (but DO include
specific games like "dwarf_fortress_player")
- Country-level locations like "lives_in_usa"
(but DO include city/region like
"lives_in_portland")
- One-time mentions or temporary states
(job_hunting, currently_sick, just_moved)

PRIORITIZE distinctive persistent traits:
- Specific locations (city/region level):
lives_in_portland, lives_in_nelson_bc
- Profession field: aerospace_engineer,
pediatric_nurse, high_school_teacher
- Core identity: gender, has_children,
homeowner, has_dog, has_cat
- Deep hobbies (not one-off mentions):
woodworking, homebrewing, keeps_reptiles
- Strong values: pro_nuclear, vegan,
libertarian, atheist, gun_owner
- Specific fandoms: dwarf_fortress_player,
formula1_fan, liverpool_fc_supporter
- Life circumstances: expat, military_veteran,
recovering_alcoholic

Format:
- Lowercase, 1-4 words, underscores between
words
- Be thorough - more traits is better than
fewer - Order by distinctiveness (most
identifying first)

Examples of GOOD output:
lives_in_portland, male, aerospace_engineer,
has_dog, woodworking_interest, pro_nuclear,
recovering_alcoholic, nfl_fan, atheist

lives_in_uk, female, nurse, has_children,
marathon_runner, plays_guitar,
labour_supporter

Examples of BAD output (too generic or
unstable):
male, age_30s, lives_in_usa, married,
employed, gaming_interest, uses_profanity,
analytical_tone, job_hunting

Now output the descriptors.
```

The following is a synthetic example summary illustrating the output format. It does not correspond to any real user.

Example Output (Synthetic)

```
lives_in_nelson_bc, pediatric_nurse,
keeps_ball_pythons, homebrews_mead,
female, married, has_two_daughters,
roller_derby_player, prius_owner,
sourdough_obsessed, plays_stardew_valley,
critical_role_fan, ndp_voter, pro_nuclear,
celiac_disease, plays_mandolin,
thru_hiked_pct, volunteer_firefighter,
dislikes_cilantro
```

Record selection. The following prompt is used with Gemini 3 Pro to perform LLM-based record selection using the top-k candidates.

Record Selection Prompt

```
You are matching Reddit users across two time
periods. Given a QUERY user profile and k
CANDIDATE profiles, determine which candidate
(if any) is the same person as the query.
```

```
QUERY profile:
query_summary

CANDIDATE profiles:
candidates

Which candidate is the same person as the
query? Consider overlapping traits like
location, profession, hobbies, demographics,
and values. A match should share multiple
distinctive traits, not just one or two common
ones.

Output ONLY a single number (1-k) for the best
match.
```

Sorting (pairwise comparisons). The following prompt is used with `gpt-5-mini-2025-08-07` to perform pairwise comparisons between query-candidate pairs for confidence sorting.

Pairwise Comparison Prompt

```
# Task
You are helping to match Reddit accounts. We
have two accounts: 'query' and 'candidate'.
The goal is to determine if a query-account
and candidate-account belong to the same
person.

You will see two matches (MATCH A and MATCH
B). Each match proposes that a specific
query-account and candidate-account are the
same person. Your job is to decide which match
is more plausible.

# Data format
Each account is summarized as a
comma-separated list of tags describing the
user (e.g., location, occupation, interests,
writing style).

<few-shot examples>

## MATCH A
Query: <candidate_a query summary>
Candidate: <candidate_a target summary>

## MATCH B Query: <candidate_b query summary>
Candidate: <candidate_b target summary>

# Response options - **A**: MATCH A is the
better match - **B**: MATCH B is the better
match
Respond with: A, B
```

G.3 Recall@Precision as a function of the fraction of matchable queries π

Let M be the number of matchable queries, N the number of non-matchable queries, and $\pi = \frac{M}{M+N}$. Define TP as the number of true positives (correct attacker guesses), FP_M as the number of false positives for matchable queries (wrong attacker guess when there exists a true match), and FP_N as the number of false positives for non-matchable queries (the attacker does not abstain for users without a true match). With this notation, the three key rates are

- True Positive Rate/Recall: $\text{TPR} = \frac{\text{TP}}{M}$
- False Match Rate: $\text{FMR} = \frac{\text{FP}_M}{M}$
- False Positive Identification Rate: $\text{FPIR} = \frac{\text{FP}_N}{N}$.

Precision can be rewritten in terms of those rates and π as follows:

$$\begin{aligned}\text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}_M+\text{FP}_N} \\ &= \frac{\pi\frac{\text{TP}}{M}}{\pi\frac{\text{TP}}{M}+\pi\frac{\text{FP}_M}{M}+(1-\pi)\frac{\text{FP}_N}{N}} \\ &= \frac{\pi\cdot\text{TPR}}{\pi\cdot\text{TPR}+\pi\cdot\text{FMR}+(1-\pi)\cdot\text{FPIR}}\,.\end{aligned}$$

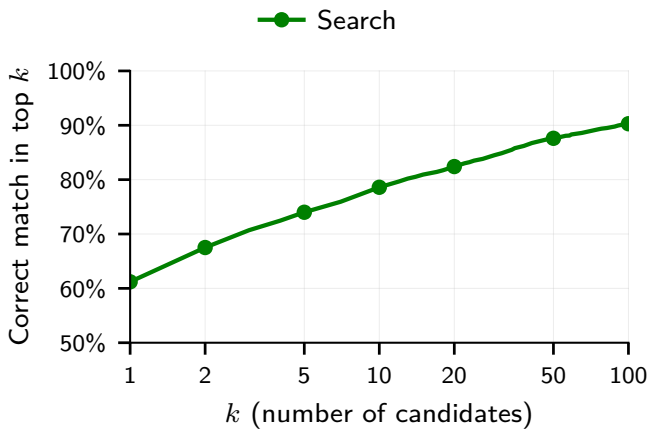
Since all three rates depend on *either* the M matchable queries or the N non-matchable queries, but *not both*, we can estimate all rates independently of π.

G.4 Justifying Reasoning and Calibration for temporally split Reddit profiles

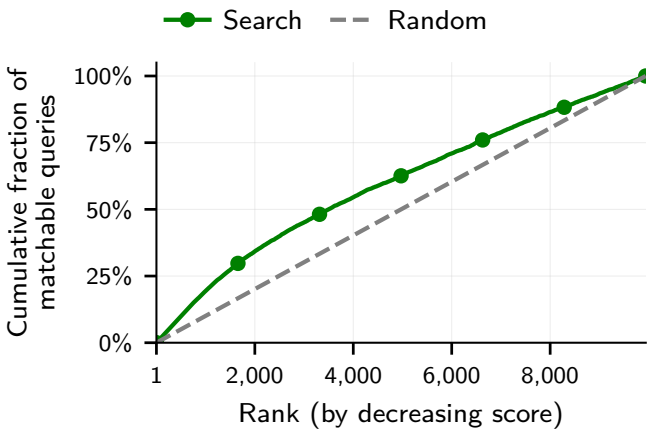
Justifying Reasoning. For every k, we plot the fraction of queries where the top-k candidates in terms of embedding similarity contain the true match in Figure 8a. While rank-1 accuracy is only around 60%, this rises to over 80% at $k = 15$. Thus, embedding similarities effectively narrow down the candidate pool to a small size (such that using LLMs with reasoning for candidate selection becomes feasible), but fail to identify the one true match consistently.

Justifying Calibration. In Figure 8b, we take the maximum query-candidate embedding similarity for all query users (5k with a true match, 5k with no true match), sort queries in decreasing similarity, and show the fraction of matchable users at every rank. The resulting curve barely surpasses random ordering. Thus, embedding similarity is poorly calibrated, justifying the use of LLMs for better calibration.

G.5 Results

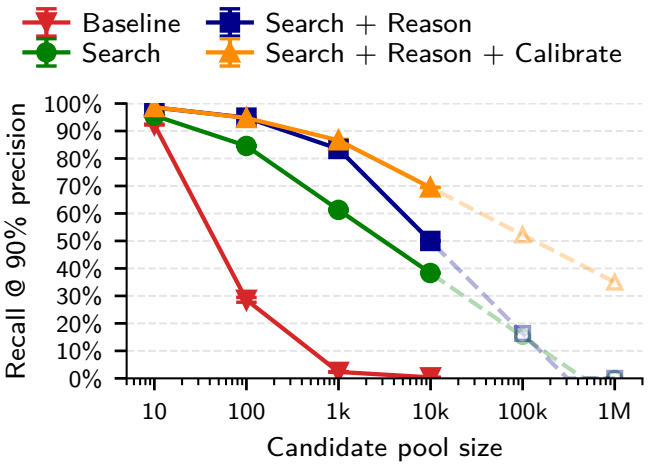


(a) Top-k recall for Search

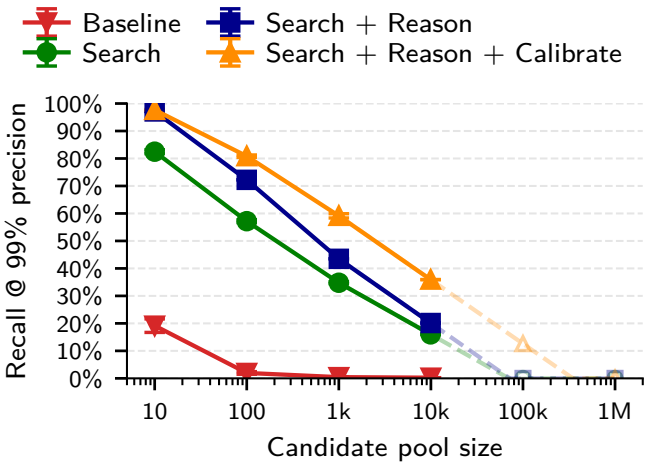


(b) Fraction of matches containing a matchable query when ranked by similarity score

Figure 8: Similarity score is good at narrowing down the candidate set (a) but not at ranking matches in terms of plausibility (b).



(a) Effects of candidate set size at 90% precision

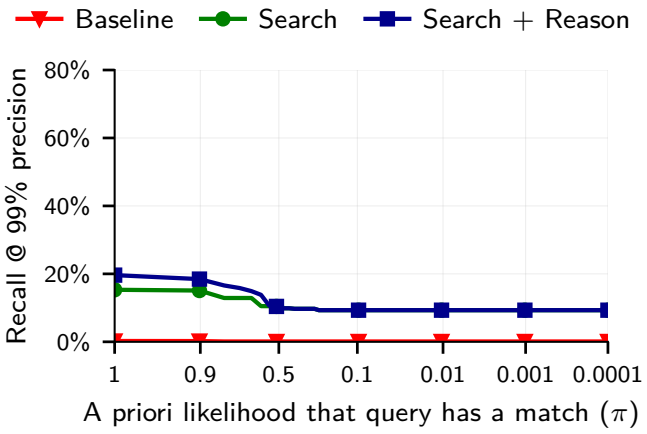


(b) Effects of candidate set size at 99% precision

Figure 9: Full results for Figure 7a.

Table 6: Recall at precision thresholds for Reddit split-profile matching (5k queries, 10k candidates).

Method	90% Prec	98% Prec	99% Prec
Baseline (Narayanan)	0.4%	0.2%	0.2%
LLM	38.3%	20.2%	16.0%
LLM + Selection	47.8%	28.2%	20.3%
LLM + Selection + Sorting	67.3%	47.6%	38.4%



(a) Effects of the likelihood that a query has a match (π) at 99% precision

Figure 10: Full results for Figure 7b.