Membership Inference in Privacy-Preserving Vector Search

Georg Fuchsbauer* Carter Luck† Adam O'Neill† Ian Rapko†

Abstract

Vector retrieval increasingly runs over sensitive embedding databases. Backend-compatible protections are desirable in such settings in that they transform the plaintext vectors while leaving the search algorithm itself unchanged. However, the protected database then leaks some geometry, that is, information about the distances between the stored vectors. We study *membership-inference attacks* on such leakage given a single database snapshot and known plaintexts. Our central finding is that membership privacy depends on how much geometry is revealed and on whether known plaintexts can be linked to positions in the snapshot, and that the effect of each of these depends on the other.

We first break the prior Scale-and-Perturb (SAP) scheme of Fuchsbauer et al. (FGHO, SCN 2022) with no known plaintexts. Empirically, our attack achieves advantage 0.997 on MS-MARCO. We then introduce Rotate-and-Perturb (RAP), a special case of a scheme FGHO call SAP-I and leave unanalyzed. For uniform data on the sphere, we prove perfect membership privacy with no known plaintexts. For fixed noise parameter $\beta \in (0, 2]$, achieving constant advantage requires a number of known plaintext-ciphertext pairs at least logarithmic in the database size, provided that number is below the embedding dimension. Empirically, our attack reaches advantage 0.5 with 3 such pairs at database size 100 and 7 at 100,000 on text embeddings.

We also study two more approaches, namely ideal *distance-comparison-revealing encryption* (DCRE) and an encrypted HNSW index using function-hiding inner-product functional encryption. Ideal DCRE remains vulnerable to known plaintext-ciphertext pairs. On the tested MS-MARCO databases, our attack on the encrypted HNSW index requires knowing 19–32% of the database and their corresponding index vertices to reach advantage 0.5.

Finally, we study augmentations to these schemes to mitigate our attacks, *shuffling* and *secret projection*. Together, they substantially increase the number of known plaintexts needed by our tested attacks on RAP and comparison-only leakage.

1 Introduction

1.1 Background and Motivation

Encrypted vector search. Vector retrieval, which is pervasive in modern machine learning systems, represents items as vectors and returns nearby items. For example, in retrieval-augmented generation (RAG), an application maps passages from a document collection to embeddings, stores them in a vector database, and retrieves passages close to an embedded query for use as context by a language model. Similar systems support semantic and code search, image retrieval, recommendation, and biometric matching [JDJ21, Wea, Qdr, Mic].

An important consideration is that the underlying collections may contain private or proprietary information, yet the vector database may be a managed service or a separate component of the same system. We consider an application that keeps a secret key outside the database and gives

*TU Wien. georg.fuchsbauer@tuwien.ac.at

†UMass Amherst. {cluck,adamoneill,irapko}@umass.edu

the database only "cryptographically protected" vectors. Depending on the approach, the entity hosting the database either builds an index from these vectors or receives an index already built by the application. The goal is to limit what a database operator or an attacker who compromises the database component can learn while keeping search both efficient and accurate.

We focus on *backend-compatible* approaches, which allow the database to use existing vector-search algorithms, such as HNSW, on protected data. While such approaches may change how distances are evaluated or require the data owner to build the index, they still allow the database to use its existing search algorithms. Backend compatibility matters not only for present performance but also because vector-search algorithms continue to improve. Retaining those algorithms lets encrypted search benefit from improvements to them without redesigning the cryptographic scheme. Production indexes already run at billion-vector scale [JDJ21, SDS$^+$19b], and retaining the usual search algorithms gives access to that engineering, along with new indexes and hardware optimizations [GSL$^+$20, CZW$^+$21, GL24, ONN$^+$24]. An approach that replaces the backend with a specialized cryptographic index or protocol has to reach that scale on its own.

In fact, encryption with a backend-compatible interface is already commercially available. IronCore Labs implements the SAP distance-comparison-preserving encryption (DCPE) scheme of [FGHO22] in the Cloaked AI vector-encryption methods of its Alloy SDK, where an application encrypts database and query embeddings before sending them to an ordinary vector database, with RAG among the intended uses [Iro24, Bue26]. One of our results is that SAP is vulnerable to membership-inference attacks, although we have not examined the implementation and make no concrete claim about their implementation.

A recent benchmark makes the case for such schemes concrete. Kermarrec et al. [KPRdV26] run SAP and three heavier alternatives on one corpus, index, and machine. They report that SAP runs at the speed of an unencrypted baseline while the other three cost about 4×, 22×, and 190× per query, and they recommend SAP when the raw contents of the documents are what needs protecting. They also observe that its ciphertexts reveal which stored items are close to one another; however, they do not ask what that geometry, the pairwise distances the ciphertexts reveal, allows an attacker to infer, as we do in this work.

Our approach. Our approach builds on *property-preserving encryption* (PPE) [BBO07, BCLO09, PR12], which allows a predicate on plaintexts to be evaluated directly from their ciphertexts, and on its extension to *function-revealing encryption* (FRE) [JP18]. Such schemes are useful in encrypted search precisely because they leave the relevant backend operation of the overlying system essentially unchanged. PPE was originally designed for SQL databases, but is widely considered insecure in that context. Attacks can recover substantial plaintext information from what such schemes reveal using a reference dataset having the same distribution as the plaintexts; see Section 1.5.

However, the above attacks do not answer the question of what privacy PPE can provide for vector databases. Here, the basic type of PPE is (approximate) *distance-comparison-preserving encryption* (DCPE) and its extension to *distance-comparison-revealing encryption* (DCRE) [FGHO22]. These preserve the ternary predicate of which of two candidate vectors is closer to an anchor vector. Distance comparisons do reveal ordering, but the difference is that they reveal a *collection of orderings*, one for each database vector, ranking the others by distance from it. Unlike order-preserving [BCLO09] and order-revealing encryption [BCO11, BLR$^+$15], which reveal a single global order, these orderings are anchored at database vectors whose plaintext identities may be *unknown*, and the whole collection depends only on how the database vectors lie relative to one another. Whether public distribution information or known database plaintexts suffice to align these orderings with plaintext vectors is precisely what we study.

Why membership inference? Rather than asking whether the database plaintexts can be recovered or whether two encrypted databases can be distinguished—as is common in the analysis of encrypted search systems—we focus on whether an adversary can determine the presence of a particular plaintext in the database. Note that partial database recovery and membership inference are incomparable in general. On the one hand, recovering some records need not determine whether a particular challenge plaintext is present; on the other hand, inferring membership of one plaintext need not lead to recovering other plaintexts. We therefore study membership inference as a separate privacy goal. Notably, membership inference is a standard metric in machine learning; see Section 1.5. We find it natural to also apply it to encrypted search systems, where the object under attack is the leakage from a scheme rather than a trained model.

Note that although the raw item rather than its embedding is what ultimately must remain private, a public or queryable embedding model lets an adversary map any data item to its embedding. We therefore study the privacy of the embeddings themselves, while treating any protection gained from a hidden model as a separate, deployment-dependent barrier.

Why snapshot security? A snapshot adversary holds a copy of the stored state; this could be the attacker who obtains, e.g., a database dump, a stolen backup, or disk access at a managed service. Snapshot security is a recognized model in work on encrypted databases, mostly in the attack setting starting with [NKW15, GSB$^+$16]. Such an adversary sees less than one who also observes queries and updates. Our attacks are stronger for this, since they need only the stored state. On the other hand, our security results are weaker for this, since an adversary who also sees queries may succeed where our attacks fail.

Indeed, going beyond the snapshot model seems hard while keeping the scheme backend-compatible as we want. A backend-compatible deployment runs the database's usual search algorithm, whose traversal of the index depends on the query and is visible to whoever runs it. Hiding that traversal means an oblivious access pattern over encrypted data, as in Boldyreva and Tang and in Compass [BT21, ZPZP25]. Both replace the backend's search procedure and cost substantially more per query. We are not aware of an approach that keeps the backend unchanged and also hides access patterns.

1.2 Threat Model

Security game and adversaries. We define privacy against *membership-inference attacks* (MIA) via a cryptographic-style security game, following the formalization of Yeom et al. [YGFJ18]. In the game, a uniform challenge bit decides whether a challenge vector p is planted at a random position in the database or the database is sampled independently of p. The adversary sees the encrypted database, the challenge p, and public information about the data distribution $\mathcal{D}$ such as a disjoint reference set, and it guesses which case occurred. It sees no retrieval queries, query results, access patterns, or volumes. Our attacks are efficient and our security theorems hold against every (even unbounded) adversary. The one place we need the adversary to be computationally bounded is Section 9, where the construction uses a function-hiding inner-product functional encryption scheme, and the leakage then comes from the index itself, not from the ciphertexts.

Known-plaintext attacks. A central contribution of the paper is to study the effect of *known database plaintexts* on membership inference attacks, which is unusual in the context of property-preserving encryption. Throughout, we use k to denote the number of database plaintexts the adversary knows. Attacks on PPE in the SQL setting [NKW15, DDC16, GSB$^+$16, CGPR15] usually need no known plaintexts, only a reference dataset drawn from the plaintext distribution, so this

parameter did not arise there. These plaintexts differ from public information about $\mathcal{D}$ because they come from the challenge database itself. What matters for a deployment is not only how many plaintexts an attack needs on a small database to reach a breakpoint, but how that number grows with the size of the database. Moreover, we consider two settings of known-plaintext attack. In the *matched known-plaintext* setting the adversary also learns which ciphertext corresponds to each known plaintext, and in the *unmatched known-plaintext* setting a random relabeling of the database view hides that matching. Abstractly, we achieve the latter via shuffling, but ultimately which setting applies from a systems perspective is application-dependent.

We note that our unmatched attacks do *not* work by recovering the matching, i.e., by reducing to the matched known-plaintext setting. When the challenge is a member it is itself one of the stored plaintexts, so one may expect the attack to first find which ciphertext it corresponds to. On the contrary, the attack keeps several plausible matchings and tests the challenge against all of them at once. At $n = 100$ and $k = 5$ the lowest-error matching is the true one in 45% of runs while the attack reaches advantage 0.60 (Appendix B.4).

How many plaintexts an adversary is likely to hold is also application-dependent. When the corpus mixes private and public material, as in retrieval-augmented generation over internal documents together with public manuals or reference works, every public document in the index is a known plaintext, and k can be a large fraction of n. When the corpus is wholly private, obtaining even a few of its plaintexts needs a separate compromise, and small k is the range that matters.

1.3 Main Results

Throughout, a *breakpoint* is the smallest number of known plaintexts at which our attack reaches advantage 0.5; we also measure false-positive rate, which is the fraction of nonmembers the attack accepts at that breakpoint, and other metrics. See Section 4.3 for the details.

Provable vs. empirical results. We have three proven results below, namely the SAP attack with a known scaling factor (Theorem 5.3), RAP with no known plaintexts (Theorem 6.3), and RAP with matched known plaintexts (Theorem 6.4). Beyond these, we test specific attacks on real embeddings, namely the SAP attack with approximate scaling factor, our attacks on RAP and on projected RAP, our attack on the comparison-only leakage profile, and our attack on the encrypted HNSW index. Provable security results on real embeddings seem hard because such data is not easy to describe; we are not aware of such results in the literature.

Attack on scale-and-perturb. We first break scale-and-perturb (SAP), the DCPE scheme of FGHO [FGHO22], which multiplies every vector by a secret scaling factor ρ and adds small noise. We break it with no known plaintexts and in $O(nd)$ time. Our proof assumes a variance-bounded Gaussian distribution, and our experiments show that the attack works on real embeddings as well. The factor ρ multiplies every vector by the same constant. The adversary knows the average coordinate variance of the plaintexts, since that is a property of $\mathcal{D}$, and it can measure that of the stored vectors, so the ratio gives the scaling factor. It then divides that factor out and tests whether any ciphertext lies near the challenge. Given the exact scaling factor, we prove for variance-bounded Gaussian distributions that advantage approaches 1 exponentially fast in the dimension. The complete attack estimates the factor, and reaches advantage 0.997 at the default MS-MARCO setting without weakening over the database sizes we test.

Rotate-and-Perturb. Our break of SAP leaves open the question of a scheme with non-trivial membership privacy. For databases of unit vectors, we introduce *Rotate-and-Perturb* (RAP), which

encrypts each vector with a secret uniform orthogonal matrix plus fresh noise from the ball of radius $\beta/4$. RAP is a special case of FGHO's SAP-I, whose security they leave unanalyzed; see Section 6 for the details. The secret matrix changes every coordinate of a vector but keeps all pairwise distances, and the noise changes each distance by at most $\beta/2$. So every distance comparison whose plaintext margin exceeds β is preserved, and what the server learns is the pairwise distances between the vectors, up to the noise. RAP's leakage profile is a noisy version of their Gram matrix, which we use for the analysis.

Attacks on RAP. With no known plaintexts and $\mathcal{D}$ uniform on the sphere, we prove perfect membership privacy against even unbounded adversaries. Our attacks attempting to exploit non-uniformity did not work at the sizes we test (Section 6.4).

On MS-MARCO at $\beta = 0.5$, we give a matched attack that reaches a breakpoint with 3 plaintexts at $n = 100$, accepting 0.86 of members and 0.20 of nonmembers. Without the matching, the attack needs 5. Across database sizes from 10^2 to 10^5 the requirement increases by about one to one and a half known plaintexts per order of magnitude. The same matched attack works on CLIP [RKH+21] embeddings of MS COCO [LMB+14] images and FaceNet [SKP15] embeddings of LFW [HRBLM07] faces (Section 6). That false-positive rate is high compared with SAP, where the attack accepts 0.003 of nonmembers. About one RAP nonmember in five passes the test, so its claims about individual records are much less reliable even though its advantage reaches the same 0.5.

Applying secret projection to RAP gives our *projected* RAP scheme. It maps into fewer dimensions and so cannot be decrypted. Indeed, we treat (secret-key, randomized) hashing rather than encryption throughout, which loses nothing, since a deployment that needs the vector back can store it under a standard encryption scheme alongside the hash value. At output dimension $m = 256$ the unmatched attack reaches a breakpoint at 18 known plaintexts rather than 5 at $n = 100$, and at 128 rather than 12 at $n = 1{,}000$. Under our fixed beam-search budget the measured gap between the breakpoints of projected and unprojected RAP therefore grows with the database. We also propose a *task-aware* variant that projects inside a publicly learned representation, improving utility but not security.

Provable security for RAP on uniform data. For data uniform on the sphere and every noise level $0 < \beta \leq 2$, we bound the advantage of every (unbounded) adversary holding $k < d$ matched known plaintexts, by a quantity that falls as the inverse of the square root of the database size and grows exponentially in k (Section 6.7). Reaching any fixed constant advantage therefore needs k at least logarithmic in n; our MS-MARCO measurements show the same qualitative growth. The proof is involved and reduces the matched game to a standalone *planted sketch problem*. Then, we bound the advantage of every distinguisher against it, again with no computational assumption. In particular, this result gives a natural distribution on which RAP has nontrivial known-plaintext security. While the scope of the result is limited, we see it as a starting point for future work on provable security for the other schemes we consider and other data distributions.

Ideal DCRE and comparison-only leakage. FGHO also define ideal DCRE, where an *arbitrary* public algorithm compares distances between ciphertexts unlike DCPE. Its leakage is, for each ciphertext, the order of the others by distance, which we call the *comparison-only* leakage profile (Section 8). While a construction of ideal DCRE remains open, our study of what the adversary can deduce via the leakage profile indicates whether it would be useful to construct.

Here, we give a rank-based attack, which ranks the known plaintexts by distance from the challenge and looks for a ciphertext whose own ordering agrees with that ranking. On MS-MARCO at $n = 100$, it reaches a breakpoint with 3 matched known plaintexts and 5 unmatched, and at

noise $\beta = 0.5$ it needs 4 and 8. Secret projection raises those numbers by more. At $m = 256$ the matched attack reaches a breakpoint at 12, while the unmatched attack does not reach one through $k = 99$, achieving advantage 0.411 there and accepting 0.45 of members and 0.04 of nonmembers. The attacks and projected results are in Section 8, and additional experiments are in Appendix D.

Owner-built encrypted HNSW index. We next study an approach in which the owner encrypts each vector under *function-hiding inner-product functional encryption* (FH-IPFE) [KLM$^+$18], an advanced form of encryption where the server can only compute inner products between query vectors and data vectors. Ranking by inner product is a standard retrieval operation in its own right, and for unit vectors it is the same as ranking by distance. FH-IPFE also leaves the server unable to build an index to enable sublinear-time search, so the owner builds an HNSW index [MY20] and sends it with the ciphertexts, as proposed by Acharya et al. [AFLO25], who left the leakage unanalyzed. Here, the ciphertexts themselves leak nothing; what leaks information is the *HNSW index* the data owner builds over them, which is a graph whose edges depend on the distances between the plaintext vectors; see Section 9.

We give a matched attack that extracts membership information from the graph. On MS-MARCO its breakpoint is between 19% and 32% of the database over the sizes we test, so the number of known plaintexts it needs grows roughly in step with the database, whereas the matched attack on RAP needs 4 to 5 known plaintexts at those database sizes. At those breakpoints it accepts 0.14 to 0.25 of nonmembers, against 0.13 to 0.33 for the matched attack on RAP. Our unmatched attack fails outright, recovering known vertices within one percent of the random-set baseline. The proof of the leakage reduction and the experimental details are in Appendix E.

Mitigations. We propose mitigations with which to augment our schemes to suppress the two sources of leakage, namely geometry revealed from a database snapshot, and links between plaintexts and their positions in it. Both mitigations apply to every scheme we consider, not only to RAP. *Shuffling*, as in FGHO, stores the ciphertexts in random order and hides the links between known plaintexts and their ciphertexts. Abstractly, this achieves the unmatched setting above, provided that identifiers, metadata, and other deployment information do not restore the link. *Secret projection* removes part of the revealed geometry, mapping a unit vector to $\mathbb{R}^m$ with a secret random projection and scaling it so that the projected vector has the same length as the input on average. We find that projection raises the number of known plaintexts an attack needs, and it makes the attack that still succeeds accept fewer nonmembers. Note that adding more noise does not substitute for either mitigation. Raising β weakens our attacks at small k, but at $\beta = 1$ the effect is not enough to prevent attacks with more known plaintexts; see Appendix B.4.

1.4 Discussion and Overall Findings

Our central finding. Throughout our results, we find that membership privacy depends on how much geometry the stored state reveals and on whether known plaintexts can be linked to positions in it, and that the effect of each of these depends on the other. This dependence differs across schemes and determines when shuffling and secret projection are needed, either alone or in combination. Sometimes only one of them is needed. Against the encrypted HNSW index, our unmatched attack completely fails, so we did not even test secret projection there. Sometimes neither is sufficient alone but the combination is. For example, against the comparison-only leakage of ideal DCRE, shuffling raises our attack's breakpoint by only 2 known plaintexts and secret random projection to $m = 256$ raises it by 9, but using both leaves our attack without a breakpoint through $k = 99$. Notably, both mitigations have a more pronounced effect for the comparison-only leakage

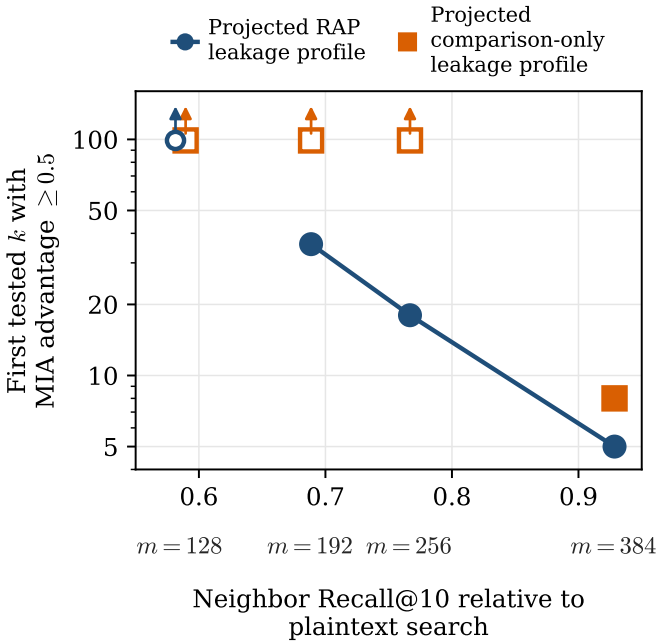

Figure 1: Attack-specific breakpoints and Neighbor Recall@10 on MS-MARCO. The two axes are measured at different database sizes. Breakpoints are unmatched known-plaintext MIA at $n = 100$ and $\beta = 0.5$, with shuffling; Neighbor Recall@10 is at $n = 10{,}000$. At the projected-RAP breakpoints for $m = 384$, 256, and 192, nonmember acceptance is 0.28, 0.16, and 0.03. At $m = 128$, where the attack reaches no breakpoint through $k = 99$, it is 0.03. Labels give projection output dimension m; $m = 384$ is unprojected. A hollow circle or square with an upward arrow means the attack does not reach advantage 0.5 through $k = 99$, the maximum tested k.

profile, which reveals less than RAP, than for RAP, where our unmatched attack on projection to $m = 256$ reaches a breakpoint at 18 known plaintexts.

The privacy–utility tradeoff. Secret projection increases the number of known plaintexts needed to reach a breakpoint, at a loss in retrieval quality. At $n = 100$ the breakpoint rises from 5 known plaintexts to 36 at $m = 192$, for about one percent of MRR@10 against the relevance judgments at $n = 10^4$. In Figure 1 we plot projected RAP and the projected comparison-only leakage profile on the same axes so that the tradeoff can be seen directly. At each output dimension m the two have the same retrieval quality, so the vertical gap between them isolates the effect of withholding the coordinates and numerical distances that RAP leaks. The rightmost point of the projected-RAP curve is RAP with shuffling alone, so moving left along that curve shows what projection adds. The stricter measure of agreement with plaintext search shows a greater cost in utility, with Neighbor Recall@10 falling from 0.929 to 0.689, because that metric counts an equally relevant substitute as a loss.

Deployment cost. Finally, we measure the performance overhead of RAP and compare it to prior work. Encrypting a query and searching the RAP index together cost a factor of 1.9 in batched query throughput on MS-MARCO and 2.7 on SciFact [WLL+20]. About a third of that gap is the encryption itself; the rest is search over the encrypted vectors. Backend compatibility keeps the

search algorithm, but it does not by itself make a scheme cheap, since the cost of each distance evaluation depends on the cryptography. Acharya et al. [AFLO25] report 7.3 to 23.5 seconds per query for their access-controlled inner-product function-revealing encryption (IPFRE) protocol, where RAP and projected RAP answer a query in 0.50 and 0.31 milliseconds, and their ciphertext is about 8 times larger per vector. The workloads and hardware are not controlled. The complete measurements are in Appendix F.3.

1.5 Related Work

Leakage-abuse attacks and encrypted search. Attacks on property-preserving encryption recover plaintexts from the equality or order a scheme permits together with auxiliary information, e.g. [NKW15, DDC16, GSB+16, CGPR15], and others reconstruct encrypted databases from volume and access patterns, beginning with Islam et al. [IKK12] and including nearest-neighbor query transcripts, e.g. [KKNO16, PWLP19, ZKP16, KPT19]. Encrypted nearest-neighbor systems built on searchable symmetric encryption and locality-sensitive hashing expose those online patterns [KIK12, YCWW15], while oblivious indexes hide them by changing the search architecture [BT21]. Compass [ZPZP25] co-designs HNSW traversal with ORAM, storing the index encrypted and hiding the access pattern of the graph walk. SANNS [CCD+20] and PACMANN [ZSF25] protect a client's query from a server that holds the database in the clear.

Left/right order-revealing encryption and CAPRISE permit comparisons between database and query encodings but not between two database encodings [LW16, YGLL26]. CAPRISE is an unbalanced version of SAP, putting more noise on database encodings than on queries, so the database-wide variance statistic our SAP attack exploits is present in it as well. Kim et al. [KLM+18] use FH-IPFE for exact nearest-neighbor search via a linear scan. Acharya et al. [AFLO25] use access-controlled inner-product function-revealing encryption (AC-IPFRE), a variant of FH-IPFE that trades leakage against efficiency differently, with an owner-built HNSW index, which in the snapshot setting gives graph-state leakage of the kind we study. None of these works studies membership inference from the encrypted database and its index.

Recovering a secret orthogonal transform. Giannella, Liu, and Kargupta [GLK13] show how to recover a secret orthogonal transform from known input–output pairs. Their setting is noise-free and asks the adversary to recover the transform itself, whereas ours is noisy and asks whether a challenge vector belongs to the database.

Graph and database alignment. Our unmatched attack searches for an assignment of known plaintexts to stored positions that agrees with all the pairwise inner products it can compute, which is the problem of recovering a hidden correspondence from correlated observations. That problem has been studied as network de-anonymization and as graph matching, starting with Pedarsani and Grossglauser [PG11], with information-theoretic thresholds for correlated random graphs [CK16] and for database alignment with Gaussian features [DCK19], efficient spectral algorithms [FMWX20], and a seeded variant in which some correspondences are given [LFP14], which is closest to our setting. Two things differ here. The observation is a noisy Gram matrix produced by a secret rotation rather than two correlated graphs. The adversary has to decide membership rather than recover the whole correspondence; indeed, our attack does not proceed by recovering it, as detailed in Appendix B.4.

Property-preserving hashing. Our DCPH notion of Section 3 is also related to a somewhat different notion called property-preserving hashing [BLV19, FS21, FLS21, HLTW22], which allows

one to evaluate predicates such as Hamming distance or set membership on preimages. Beginning with Boyle et al. [BLV19], this line of work considers adversarially chosen inputs, whereas DCPH fixes the predicate to distance comparison and we ask the adversary to determine whether a challenge vector was included in the database before hashing. Neither requirement implies the other.

Privacy-preserving nearest-neighbor systems. Liu et al. [LZX+25] build a single-server private nearest-neighbor system whose filter phase is an HNSW index over SAP ciphertexts, and whose refine phase uses a separate scheme of their own for exact distance comparisons. They adopt SAP because it keeps the backend search intact, which is the property we study here, so our SAP result applies to the filter phase of that design. Cui et al. [CLDW26] instead perturb binary codes with randomized response and search a multi-graph HNSW index over them. Their guarantee is a differential-privacy relaxation rather than an oblivious one, and covers the access and search patterns that our snapshot model does not consider.

Membership inference. Membership inference predates its use against trained models. Homer et al. [HSR+08] detect whether an individual contributed to a genomic study from published aggregate statistics alone, and Sankararaman et al. [SOJH09] formulate that attack as a likelihood-ratio test and characterize its power. The same question has since been asked of other released objects, such as genomic beacons [SB15], aggregate location data [PTC18], and synthetic datasets [SOT22]. Two prior formalizations are closest to ours. Pyrgelis et al. [PTC18] pose the question as a game over a published aggregate, so their adversary, like ours, sees a static object and no trained model. Yeom et al. [YGFJ18] give the game we use, in which a challenge bit decides whether the challenge is planted in the data and the adversary guesses that bit.

Against trained models the attack was introduced by Shokri et al. [SSSS17] and formalized by Yeom et al. [YGFJ18], and Hu et al. [HSS+22] survey that literature. Sablayrolles et al. [SDS+19a] derive the optimal membership test under a posterior model on the trained parameters. How to measure such an attack has itself been debated. Carlini et al. [CCN+22] argue that average-case metrics hide whether an attack can confidently identify any members, and evaluate at low false-positive rates instead; Watson et al. [WGCS22] calibrate difficulty per record. Jayaraman et al. [JWK+21] argue that advantage alone misstates the risk when members are rare among an adversary's candidates, which is why we report the two acceptance rates.

Zeng et al. [ZZH+24] study extraction from RAG retrieval databases, and several works study membership inference through a RAG system, e.g. [LLWY25, WHL+25, AAG25, NPS+25, LZL25]. Those attacks query a plaintext datastore through its retrieval interface. Our membership-inference game asks the same member versus nonmember question from one snapshot of the protected database and its index, without generated outputs or a query transcript.

Embedding privacy. Work on embedding inversion shows that an exposed embedding may reveal or permit reconstruction of its input [SR20, MKSR23, LXS23]. This is a different attack goal, but it motivates cryptographic protection for the stored embeddings themselves, which is what we study. These attacks query the encoder that produced the embeddings. Huang et al. [HTH+24] remove that requirement, training a surrogate model in place of querying the victim. Jha et al. [JZSM25] need no encoder at all. Given only an unpaired set of embeddings from each of two spaces, they translate between the spaces and recover the matching on over 8,000 shuffled embeddings, which is the correspondence our unmatched setting hides. Their method trains on large corpora and uses the semantic structure that both spaces learned from data. The image space of RAP is a random orthogonal image of the input space plus noise, so that structure is preserved up to the

noise and their method applies in principle. However, our own attempts to align the image space with public data did not succeed on a database of 2,000 vectors with reference samples of up to 6,000 (Section 6.4).

2 Preliminaries

Notation. We write $[d] = \{1, \ldots, d\}$ and $\mathsf{Perm}(n)$ for the set of all permutations of $[n]$. For $x \in \mathbb{R}^d$ we write $x[i]$ for the i-th coordinate. We denote the Euclidean norm by $\|\cdot\|$, the ℓ_∞ norm by $\|\cdot\|_\infty$, and the closed Euclidean ball of radius r around p by $\mathcal{B}(p; r)$. For $x, y \in \mathbb{R}^d$ we write $\mathsf{dist}(x, y) := \|x - y\|$ for Euclidean distance. For random vectors X we write $\mathbb{E}[X]$ for expectation and $\mathrm{Cov}(X)$ for the covariance matrix; $\mathrm{Var}(X)$ is scalar variance when X is scalar. We use $\langle \cdot, \cdot \rangle$ for the standard inner product on $\mathbb{R}^d$. For unit vectors $\|x - y\|^2 = 2 - 2\langle x, y \rangle$, so distance and inner product determine each other, and ranking by increasing distance is the same as ranking by decreasing inner product. When random variables X and Y on a common space have the same distribution, we write $X \equiv Y$.

Terminology. A *plaintext vector* is a database vector before it is hashed or encrypted, a *hash value* is the output of a distance-comparison-preserving hash (DCPH), defined in Section 3, and a *ciphertext* is the output of an encryption scheme; *encoding* covers either output. We allow hash functions to be secret-key and randomized.

A *database* $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$ is an ordered tuple of vectors. We also write $\boldsymbol{x}$ for a database, which is the notation Section 4 uses. We write $\mathcal{D}^n$ for n i.i.d. samples from $\mathcal{D}$ and $x \leftarrow\!\!\$\ \mathcal{D}$ for one sample. All distributions are over $\mathbb{R}^d$ unless stated otherwise.

Gram matrices. For an ordered database $\boldsymbol{x} = (x_1, \ldots, x_n)$, its *Gram matrix* is

$$\Gamma(\boldsymbol{x}) := \big(\langle x_i, x_j \rangle\big)_{i,j \in [n]}. \tag{1}$$

Since $\mathsf{dist}(x_i, x_j)^2 = \Gamma_{ii} + \Gamma_{jj} - 2\Gamma_{ij}$, it records the database's full geometry while leaving its coordinate frame unspecified.

Orthogonal matrices and the uniform distribution on the sphere. We write $O(d)$ for the set of $d \times d$ matrices M satisfying $M^\top M = I$, the *orthogonal group*. Its elements are called orthogonal matrices; equivalently, their columns form an orthonormal basis. Multiplication by an element of $O(d)$ leaves inner products and Euclidean distances unchanged, since $\langle Mx, My \rangle = x^\top M^\top M y = \langle x, y \rangle$. We write $M \leftarrow\!\!\$\ O(d)$ for a uniformly random orthogonal matrix, which satisfies $UM \equiv M$ and $MU \equiv M$ for every fixed $U \in O(d)$. For $U \in O(d)$ and a tuple $\boldsymbol{x} = (x_1, \ldots, x_k) \in (\mathbb{R}^d)^k$ write $U\boldsymbol{x} = (Ux_1, \ldots, Ux_k)$, and for a distribution $\mathcal{D}$ on $(\mathbb{R}^d)^k$ write $U\mathcal{D}$ for $U\boldsymbol{X}$ when $\boldsymbol{X} \sim \mathcal{D}$. We call $\mathcal{D}$ *rotation invariant* if $U\mathcal{D} \equiv \mathcal{D}$ for every fixed $U \in O(d)$. We say that a random tuple is rotation invariant as shorthand for its distribution being so. We write $x \leftarrow\!\!\$\ S^{d-1}$ for a uniform sample from the sphere. We use that if $v \in S^{d-1}$ is fixed and $M \leftarrow\!\!\$\ O(d)$, then $Mv \leftarrow\!\!\$\ S^{d-1}$, and the uniform distribution is the only rotation-invariant distribution on S^{d-1}.

Datasets and encoders. We use MS-MARCO passages and queries [NRS+16], SciFact abstracts, claims, and relevance judgments [WLL+20, TRR+21], MS COCO images [LMB+14], and LFW face images [HRBLM07], together with synthetic distributions. Text is embedded with public Sentence-Transformers models [RG19], images with CLIP [RKH+21], and faces with FaceNet [SKP15]. Every

embedding is normalized to unit length, so the distribution $\mathcal{D}$ our games sample from depends on both the dataset and the encoder. Attack strength and retrieval quality are not measured at the same database sizes, so a reader comparing an attack number with a retrieval number should check which sizes each came from. The sizes used in each experiment, together with builds, dimensions, and provenance, are in Appendix F.1.

Utility metrics. Retrieval metrics are computed on a fixed number of top-ranked results, and the suffix @k names that number, so MRR@10 is computed on the first ten results. Neighbor Recall@k is the fraction of the plaintext top-k set retained after hashing. Mean reciprocal rank, MRR@10, is computed against human relevance judgments rather than against plaintext search, and is the average over queries of one divided by the rank of the first relevant result. The remaining retrieval metrics are defined in Appendix F.1.

3 Distance-Comparison-Preserving Hashing

3.1 DCPH Schemes

A distance-comparison-preserving hashing (DCPH) scheme is a DCPE scheme without decryption. A deployment can store the plaintext vector under a standard encryption scheme alongside the hash value, so up to this cost our restriction is without loss of generality.

A DCPH scheme with input dimension d and output dimension d_h is a keyed hashing scheme $\mathcal{H} = (\mathsf{Kg}, \mathsf{H})$ whose key-generation algorithm outputs a key K and whose hashing algorithm maps $\mathbb{R}^d$ to $\mathbb{R}^{d_h}$. We allow H to be randomized. Both spaces use Euclidean distance, and below dist denotes the distance in the space containing its arguments. Applied component-wise to a database tuple $S = (x_1, \ldots, x_n)$, we write

$$\mathsf{H}_K(S) = (\mathsf{H}_K(x_1), \ldots, \mathsf{H}_K(x_n)).$$

If H is randomized, the entries of $\mathsf{H}_K(S)$ are generated by independent evaluations of H_K.

The correctness requirement is that hash values preserve the distance comparisons needed for retrieval. In the exact case, if x is closer to y than to z, then $\mathsf{H}_K(x)$ is closer to $\mathsf{H}_K(y)$ than to $\mathsf{H}_K(z)$. We model tolerated ranking error with a comparison slack γ. Formally, for $\gamma \geq 0$, $\mathcal{H}$ is a γ-DCPH scheme if, for every key K in the support of Kg, all vectors x, y, z, and every possible choice of hash coins,

$$\mathsf{dist}(x, y) < \mathsf{dist}(x, z) - \gamma \quad \Longrightarrow \quad \mathsf{dist}(h_x, h_y) < \mathsf{dist}(h_x, h_z),$$

where $h_x \leftarrow\!\!\$\ \mathsf{H}_K(x)$, $h_y \leftarrow\!\!\$\ \mathsf{H}_K(y)$, and $h_z \leftarrow\!\!\$\ \mathsf{H}_K(z)$. The same value h_x is used in both distances. The slack γ is a margin on the input-vector distances that the hashing is allowed to reorder. It is a correctness parameter, not necessarily a noise parameter. For SAP below, the construction parameter β determines this slack.

3.2 Scale-and-Perturb

We adapt the no-decryption randomized-hash core of the SAP scheme of FGHO [FGHO22] to the continuous domain $\mathbb{R}^d$ and the key distribution used in our experiments. Its public parameters are $\beta \in \mathbb{R}_{\geq 0}$, the noise size, and $0 < \rho_{\min} < \rho_{\max}$, the range of the secret scaling factor ρ. Key generation samples ρ uniformly from $[\rho_{\min}, \rho_{\max}]$. On input $x \in \mathbb{R}^d$, the hash algorithm samples fresh noise η uniformly from the ℓ_2 ball of radius $\rho\beta/4$ and returns

$$\mathsf{H}_\rho(x) = \rho x + \eta.$$

Figure 2 states the scheme as pseudocode.

Hash function SAP (Scale-and-Perturb)

Parameters: noise $\beta \geq 0$ and scaling-factor range $[\rho_{\min}, \rho_{\max}]$.

Kg():

1. Sample $\rho \leftarrow\!\!\$\ \mathcal{U}([\rho_{\min},\ \rho_{\max}])$. // secret scaling factor
2. Return $K = \rho$.

$\mathsf{H}_\rho(x)$: Input: $x \in \mathbb{R}^d$.

1. Sample η uniformly from the ℓ_2 ball of radius $\rho\beta/4$. // fresh per evaluation
2. Return $h \leftarrow \rho\, x + \eta$.

Figure 2: The no-decryption scale-and-perturb hashing scheme used in this paper.

Remark 3.1 (SAP has correctness slack β). For $h_x = \rho x + \eta_x$, $h_y = \rho y + \eta_y$, and $h_z = \rho z + \eta_z$, and for $w \in \{y, z\}$,

$$\big|\mathsf{dist}(h_x, h_w) - \rho\mathsf{dist}(x, w)\big| \leq \|\eta_x - \eta_w\| \leq \rho\beta/2.$$

Each of the two hashed distances $\mathsf{dist}(h_x, h_y)$ and $\mathsf{dist}(h_x, h_z)$ therefore differs from its scaled plaintext distance by at most $\rho\beta/2$. If $\mathsf{dist}(x, y) < \mathsf{dist}(x, z) - \beta$, then

$$\begin{aligned}\mathsf{dist}(h_x, h_z) - \mathsf{dist}(h_x, h_y) &\geq \rho\big(\mathsf{dist}(x, z) - \mathsf{dist}(x, y)\big) - \rho\beta \\ &> 0,\end{aligned}$$

so SAP is a β-DCPH scheme.

4 Membership-Inference Security

4.1 Snapshot Leakage and the MIA Game

Throughout, we represent a database as a tuple $\boldsymbol{x} = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$.

Definition 4.1 (Snapshot leakage profile)**.** Fix public parameters θ. A *snapshot leakage profile* $\mathcal{L}_\theta$ is a randomized algorithm that, on input a database $\boldsymbol{x} \in (\mathbb{R}^d)^n$, outputs a value in some output space.

The database size n, the dimension d, and the public parameters θ are fixed and known to the adversary, so we do not list them as inputs below.

Definition 4.2 (Simulation and equivalence of leakage profiles)**.** Let $\mathcal{L}_\theta$ and $\mathcal{L}'_\theta$ be leakage profiles for the same database domain and public parameters. We say that $\mathcal{L}'_\theta$ is *simulatable from* $\mathcal{L}_\theta$ if there is a randomized algorithm Sim, running in time polynomial in n and d, such that

$$\mathsf{Sim}\big(\mathcal{L}_\theta(\boldsymbol{x})\big) \equiv \mathcal{L}'_\theta(\boldsymbol{x})$$

for every database $\boldsymbol{x} \in (\mathbb{R}^d)^n$, where Sim uses coins independent of those of $\mathcal{L}_\theta$. We say that $\mathcal{L}_\theta$ and $\mathcal{L}'_\theta$ are *equivalent* if each is simulatable from the other.

Simulation here refers to the abstract leakage values defined by our model, not to serialization, memory layout, timing, or other implementation artifacts, which are outside the leakage profile unless we include them.

If $\mathcal{L}'_\theta$ is simulatable from $\mathcal{L}_\theta$, then $\mathcal{L}_\theta$ is the more revealing of the two. An attack on $\mathcal{L}'_\theta$ then gives an attack on $\mathcal{L}_\theta$, since the adversary can run the simulator itself, and a security bound for $\mathcal{L}_\theta$ gives one for $\mathcal{L}'_\theta$. Equivalence means that both these directions hold.

Definition 4.3 (Trivial leakage profile of a DCPH scheme)**.** Let $\mathcal{H} = (\mathsf{Kg}, \mathsf{H})$ be a DCPH scheme. Its *trivial leakage profile* $\mathcal{L}^{\mathrm{triv}}_{\mathcal{H}}$, on input $\boldsymbol{x} = (x_1, \ldots, x_n)$, samples $K \leftarrow\!\!\$\ \mathsf{Kg}$, independently computes $h_i \leftarrow\!\!\$\ \mathsf{H}_K(x_i)$ for every $i \in [n]$, and outputs

$$\mathcal{L}^{\mathrm{triv}}_{\mathcal{H}}(\boldsymbol{x}) = (h_1, \ldots, h_n).$$

The same definition applies to any keyed scheme that produces one encoding per record, with H_K replaced by that scheme's algorithm.

We will mainly analyze simpler, abstract leakage profiles that are equivalent to the trivial leakage profile for a particular scheme.

Definition 4.4 (MIA game and advantage)**.** Let $\mathcal{L}_\theta$ be a snapshot leakage profile on databases in $\mathbb{R}^d$, let $\mathcal{D}$ be a distribution over $\mathbb{R}^d$, and let $n \geq 1$. Define the following game:

$\mathbf{Game}^{\mathsf{MIA}}_{\mathcal{L}_\theta,\mathcal{D},n}(\mathcal{A})$
$b \leftarrow\!\!\$\ \{0,1\}$
$(x_1, \ldots, x_n) \leftarrow\!\!\$\ \mathcal{D}^n \ \ ; \ \ p \leftarrow\!\!\$\ \mathcal{D}$
$j \leftarrow\!\!\$\ \{1, \ldots, n\}$
If $b = 1$: $x_j \leftarrow p$
$Y \leftarrow\!\!\$\ \mathcal{L}_\theta(x_1, \ldots, x_n)$
$b' \leftarrow \mathcal{A}((n, d, \theta), Y, p)$
Return $[b = b']$

The MIA-*advantage* of $\mathcal{A}$ is

$$\mathbf{Adv}^{\mathsf{MIA}}_{\mathcal{L}_\theta,\mathcal{D},n}(\mathcal{A}) = 2\Pr\Big[\mathbf{Game}^{\mathsf{MIA}}_{\mathcal{L}_\theta,\mathcal{D},n}(\mathcal{A}) \Rightarrow 1\Big] - 1.$$

When $\mathcal{L}_\theta = \mathcal{L}^{\mathrm{triv}}_{\mathcal{H}}$ for a DCPH scheme $\mathcal{H}$, we write $\mathbf{Game}^{\mathsf{MIA}}_{\mathcal{H},\mathcal{D},n}$ and $\mathbf{Adv}^{\mathsf{MIA}}_{\mathcal{H},\mathcal{D},n}$ as shorthand.

Discussion. Write TPR for the probability that the adversary outputs 1 when $b = 1$, and FPR for the probability that it outputs 1 when $b = 0$. Since the challenge bit is uniformly random, success probability is $\frac{1}{2}\mathrm{TPR} + \frac{1}{2}(1 - \mathrm{FPR})$, so the advantage equals $\mathrm{TPR} - \mathrm{FPR}$. We assume wlog. advantage is positive. We use advantage 0.5 throughout as a common empirical reference point; as discussed in Section 4.3, it is not a fixed boundary between secure and insecure. It corresponds to guessing the challenge bit correctly three quarters of the time in the game.

We use *member* and *nonmember* as shorthand for the planted ($b = 1$) and independently sampled ($b = 0$) cases. We call TPR and FPR the *member* and *nonmember acceptance rates*. Note that the adversary receives no retrieval queries, query results, access patterns, result volumes, or updates, which is what makes this a snapshot notion. As the challenge p is drawn from the same distribution $\mathcal{D}$ as the database entries, p on its own gives no information about b. Any advantage comes from how the leakage Y relates to p.

The game samples the database independently from $\mathcal{D}$, so all our results are stated in that model. Some results place additional requirements on $\mathcal{D}$. Theorem 5.3 takes $\mathcal{D}$ to be an ℓ-variance-bounded multivariate normal distribution, and Theorem 6.3 uses uniform random data on the sphere. Our attacks need no assumption on $\mathcal{D}$ to run, and we evaluate them on real embeddings drawn without replacement from a fixed pool, which are not independent samples from $\mathcal{D}$.

Comparison to prior notions. Fuchsbauer *et al.* [FGHO22] prove security for SAP in a *real-or-replaced game*, where the adversary submits a database and must tell whether the encrypted database is that one or the same one with one entry moved a short distance. That game and Definition 4.4 are not directly comparable from their definitions. Their adversary knows the whole database while ours knows at most part of it, so in that regard theirs is stronger. But, for them, the two cases differ in one entry that *barely moved* while ours differ in whether the challenge is in the database at all. A database sampled independently of the challenge usually contains no vector near it, as Lemma 5.2 makes precise. Our attacks therefore do not contradict their results. Our MIA notion is closer to that of Yeom et al. [YGFJ18].

4.2 Matched and Unmatched Attacks

An adversary may know some of the vectors actually stored in the challenge database, not only their distribution. We call this a *known-plaintext attack*. This happens, for example, when a collection mixes public and private records, or when part of a corpus has already been published.

Matched vs unmatched. Knowing a vector is in an encrypted database is not the same as knowing where it is in the visible stored state. If the adversary can associate a known record with the stored item containing its encoding, we call this the *matched* known-plaintext attack setting; otherwise we call it the *unmatched* setting. Thus the property separating the two settings is *record–encoding unlinkability*. Throughout the paper, we use *shuffling* as convenient shorthand for this property. Whether a deployment achieves this is application-dependent. Identifiers returned with a result, filter metadata, insertion order, and update logs can each reintroduce the correspondence. Shuffling as we defined it only makes sense when the database is static, known up front by the data owner. For dynamic applications it seems difficult to hide links between known plaintexts and their encodings, further underscoring the relevance of our study of the matched setting.

The definition. We next define MIA when the adversary knows k vectors in the challenge database. Let $\mathcal{L}_\theta$ be a snapshot leakage profile, let $\boldsymbol{x} = (x_1, \ldots, x_n)$, and for $\pi \in \mathsf{Perm}(n)$ let $\boldsymbol{x}^\pi = (x_{\pi^{-1}(1)}, \ldots, x_{\pi^{-1}(n)})$, so that π places x_i at position $\pi(i)$. For $r \in \{\mathsf{match}, \mathsf{unmatch}\}$ and $0 \le k < n$, $\mathsf{Shuf}_k^r[\mathcal{L}_\theta](\boldsymbol{x})$ samples $\pi \leftarrow\!\!\$\ \mathsf{Perm}(n)$ and $Y' \leftarrow\!\!\$\ \mathcal{L}_\theta(\boldsymbol{x}^\pi)$, then returns

$$\begin{cases} \big(Y', (x_i, \pi(i))_{i=1}^k\big), & r = \mathsf{match}, \\ \big(Y', (x_i)_{i=1}^k\big), & r = \mathsf{unmatch}. \end{cases}$$

The leakage profile runs on the shuffled database $\boldsymbol{x}^\pi$, and $\pi(i)$ is the index that the known vector x_i occupies in that database. We call the true map $i \mapsto \pi(i)$ from known plaintexts to their positions the *matching*; an attack's candidate for it is an *assignment*.

Every leakage profile we define labels its output by position, and a position names a stored item without saying which plaintext produced it.

Definition 4.5 (Matched and Unmatched MIA)**.** Let $\mathcal{L}_\theta$ be a snapshot leakage profile on databases in $\mathbb{R}^d$, let $r \in \{\mathsf{match}, \mathsf{unmatch}\}$, let $\mathcal{D}$ be a distribution over $\mathbb{R}^d$, and let $n \geq 1$ and $0 \leq k < n$.

The game $\mathbf{Game}^{\mathsf{MIA}^r}_{\mathcal{L}_\theta,\mathcal{D},n,k}(\mathcal{A})$ is the game of Definition 4.4 with two changes. The challenge position j is drawn from $\{k+1,\ldots,n\}$ rather than from $\{1,\ldots,n\}$, and the adversary receives $\mathsf{Shuf}^r_k[\mathcal{L}_\theta](x_1,\ldots,x_n)$ in place of $\mathcal{L}_\theta(x_1,\ldots,x_n)$.
Define

$$\mathbf{Adv}^{\mathsf{MIA}^r}_{\mathcal{L}_\theta,\mathcal{D},n,k}(\mathcal{A}) = 2\Pr\Big[\mathbf{Game}^{\mathsf{MIA}^r}_{\mathcal{L}_\theta,\mathcal{D},n,k}(\mathcal{A}) \Rightarrow 1\Big] - 1.$$

Taking $k = 0$ in either the matched or unmatched game gives a game distributionally equivalent to the basic MIA game of Definition 4.4, for any leakage profile, because the database tuple is exchangeable given either challenge bit and an extra independent shuffle therefore changes nothing.

We stress that the known database vectors are the first k vectors $x_1,\ldots,x_k$, sampled i.i.d. from $\mathcal{D}$ together with the rest of the database. The adversary does not choose them, either in advance or adaptively after seeing its input, and the challenge index is chosen outside $\{1,\ldots,k\}$ so that the planted challenge is not among them. A stronger adversary would select which stored records it knows, picking the ones whose geometry is most informative, and this needs no write access because the records are already stored. We investigate such an adversary empirically in Appendix F.2. A stronger one still would inject records of its own choosing, which does need write access to the corpus. That third capability is the *chosen*-plaintext setting, which our model excludes.

Simulation between leakage profiles still applies in both games. These games run the leakage profile on a shuffled database, and Definition 4.2 asks for simulation on every database, so a shuffled database is covered like any other.

Comparison to differential privacy. Differential privacy [DMNS06] is the usual notion for membership inference elsewhere, and the known-plaintext parameter is one thing that separates it from ours. A differentially private guarantee protects a record even when the adversary knows every other record and knows which record is in question. The closest point in our parameterization is the matched game with $k = n-1$. That is still a weaker demand, because our known records are drawn from $\mathcal{D}$ while differential privacy is worst case over them as well. We treat k as a parameter because a deployment usually faces a small k, and our attacks reach a breakpoint far below $n-1$.

Differential privacy also quantifies over every pair of neighboring databases and assumes nothing about where the data comes from. Our games sample the database from $\mathcal{D}$ and average over that draw. Our statements are therefore about a distribution, not about a scheme alone. Definitions that draw privacy from the randomness of the data itself, and that name the adversary's side information explicitly, are closer to ours in this respect [BGKS13, BBG+11]. Li et al. [LQS+13] show that differential privacy is itself membership privacy against a particular family of adversary priors.

A β-DCPH scheme cannot be differentially private whenever its domain contains two databases that differ in one record and disagree on a distance comparison with plaintext margin greater than β. Correctness puts the leakage of those two databases into disjoint sets, so a distinguisher wins with probability one, and no finite ε with $\delta < 1$ is possible. On the unrestricted domain such a pair exists for every β. On unit vectors the margin is at most 2, so the argument covers $\beta < 2$, and it does not apply to projected RAP, which has no such worst-case comparison guarantee. This is why we state a distributional notion rather than an (ε, δ) guarantee.

Differential privacy and our notion can both be stated in terms of the two error rates of a membership test. An (ε, δ) guarantee is equivalent to a bound on the trade-off between them, namely $\mathrm{TPR} \leq e^\varepsilon \mathrm{FPR} + \delta$ [KOV15], and f-differential privacy makes that trade-off the primitive [DRS22]. For our attacks, the acceptance rates we report give one point $(\mathrm{FPR}, \mathrm{TPR})$ of that trade-off, measured rather than guaranteed, and the advantage $|\mathrm{TPR} - \mathrm{FPR}|$ is computed from that point.

4.3 Attack Metrics

Our attacks generally produce a real-valued score and then compare it with a threshold chosen on separate calibration runs, and which side of the threshold means member is fixed there too. Performance is measured on disjoint held-out runs.

Advantage. Definition 4.4 gives an attack advantage $\mathrm{TPR} - \mathrm{FPR}$. Reversing its guesses negates this difference, so the better of the two orientations has advantage $|\mathrm{TPR} - \mathrm{FPR}|$. We report this absolute difference. Advantage is our primary quantitative metric of attack success and the quantity our theoretical results bound. Empirically, each held-out run is paired, meaning that we score one member and one nonmember on the same database and randomness. We average the signed acceptance-rate difference over seeds and take the absolute value once, at the end; see Appendix F.1.

The largest advantage any adversary achieves equals the statistical distance between the distributions of the adversary's input when $b = 1$ and when $b = 0$. A bound of ε therefore gives $\mathrm{TPR} \leq \mathrm{FPR} + \varepsilon$ at every threshold, including thresholds where the FPR is very small. An advantage bound constrains the whole trade-off curve rather than its average behavior alone.

Acceptance rates. Usually, cryptographic security means the absence of an attack achieving non-negligible advantage. In membership inference, however, the practical effect of constant advantage also depends on how credible the attack's positive membership claims are [CCN+22, JWK+21]. This matters here because several of our attacks achieve constant advantage. For example, member and nonmember acceptance rates $(0.81, 0.25)$ and $(0.57, 0.01)$ both give advantage 0.56, but on a pool of 1000 candidates holding 10 members the first accepts about 248 nonmembers while the second accepts about 10, so two attacks with the same advantage can have very different consequences for an adversary acting on a positive membership claim. We therefore report the member and nonmember acceptance rates wherever the threshold is fixed in advance.

AUC. We use AUC as an auxiliary metric to analyze our attacks. To define it, run the game once with $b = 1$ and once with $b = 0$, and orient the score so that larger values mean member, as the calibration runs fix it. AUC is the probability that the member score exceeds the nonmember score, counting ties as half, and a score independent of membership gives $\mathrm{AUC} = 1/2$. We pool scores across runs, so this compares every member score with every nonmember score and not only with the nonmember scored on the same database. AUC measures ranking rather than any one threshold, so low advantage together with high AUC points to a calibration problem rather than to leakage that gives little membership information.

Breakpoints. Where we sweep the number k of known plaintexts, the *breakpoint* is the smallest tested k at which the attack reaches MIA advantage 0.5, as in Definition 4.5. The value 0.5 is arbitrary and serves only as a common reference point. An attack below its breakpoint can still be substantial, especially when its nonmember acceptance rate is small, so we report the largest tested k below each breakpoint and, for calibrated attacks, the two acceptance rates.

5 Attacks on the Scale-and-Perturb Scheme

In this section, we break the no-decryption randomized-hash form of Scale-and-Perturb (SAP) recalled in Figure 2, under our membership-inference notion and without known plaintexts. The same attack applies irrespective of whether SAP is decryptable. We are not contradicting a theorem of

FGHO, and we do not identify an error in their analysis. They prove SAP secure in a real-or-replaced game, which, as explained in Section 4, is not directly comparable to Definition 4.4.

5.1 Membership-Inference Attack on SAP

The attack has two parts. The adversary first calls a subroutine Getr that estimates the secret scaling factor from the hashed database. It then divides that estimate out and tests a distance threshold.

Variables and notation. We write H for the hashed database, p for the challenge, and ε for the acceptance threshold, the attack's one parameter. A *bar* denotes a quantity fixed by the distribution and the scheme parameters and a *hat* denotes an estimate computed from the hashed database, so $\hat{\rho}$ estimates ρ and $\hat{v}_h$ estimates $\bar{v}_h$, the average variance of a hash-value coordinate. Two further constants appear, ℓ from Definition 5.1 and $c_{\varepsilon,\ell}$ from Lemma 5.2.

The adversary. In the following, $\mathrm{Var}_{x\sim\mathcal{D}}(x[i])$ is the variance of the i-th coordinate of a plaintext, and $\bar{\sigma}_x^2$ is the average of these d numbers,

$$\bar{\sigma}_x^2 \;=\; \frac{1}{d}\sum_{i=1}^{d} \mathrm{Var}_{x\sim\mathcal{D}}(x[i]).$$

This is a property of $\mathcal{D}$ alone, which the adversary may depend on, so the adversary can use it. The scaling factor is the secret key and the adversary can learn about it only from the hashed database, which is what Getr does: on input $\bar{\sigma}_x^2$, the public parameter β, and the hashed database H, it returns an estimate $\hat{\rho}$ of ρ. Once the adversary has $\hat{\rho} \approx \rho$, dividing each hash value by $\hat{\rho}$ recovers the corresponding plaintext up to the noise. The adversary therefore scans the rescaled values and accepts when one of them lies within a threshold $\varepsilon > 0$ of the challenge p, where ε is a parameter of the attack. We define $\mathcal{A}_{\mathcal{D}}^{\mathsf{MIA}}$ as follows:

$\mathcal{A}_{\mathcal{D}}^{\mathsf{MIA}}(H,\, p)$
$\hat{\rho} \leftarrow \mathsf{Getr}(\bar{\sigma}_x^2, \beta, H)$ For $y \in H$: If $\|y/\hat{\rho} - p\| \le \varepsilon$ then return 1 Return 0

Since both Getr and the membership test make only a constant number of passes over the n hash values and inspect d coordinates per value, the complete attack runs in $O(nd)$ time.

Analysis with exact scaling factor. We first analyze the adversary in an idealized setting where it is given the secret key, so that Getr always returns $\hat{\rho} = \rho$. Because of the noise, the adversary cannot recover the exact plaintexts, but it can still determine whether a given challenge is in the database with high probability. A member's rescaled hash value lies within $\beta/4$ of the challenge, so the test accepts. A nonmember is rejected unless some database vector happens to lie within 2ε of the challenge. Bounding that chance needs a condition on $\mathcal{D}$. We call a multivariate normal distribution *ℓ-variance bounded* if every coordinate keeps variance at least ℓ after conditioning on the preceding ones. For such a distribution the chance that one database vector lies within 2ε of the challenge in ℓ_∞ is less than $(c_{\varepsilon,\ell})^d$, for a constant $c_{\varepsilon,\ell} \in (0,1)$ that does not depend on d.

Definition 5.1 (ℓ-Variance Bounded)**.** Let $q \sim \mathcal{N}(\mu, \Sigma)$ in $\mathbb{R}^d$ and let $\ell > 0$. We call its law *ℓ-variance bounded* if, for every $0 \le r < d$, the regular conditional variance of coordinate $r+1$ given the first r coordinates is at least ℓ. Equivalently, for $r > 0$ the corresponding Schur complement

$$\Sigma_{r+1,r+1} - \Sigma_{r+1,1:r}\Sigma_{1:r,1:r}^{-1}\Sigma_{1:r,r+1} \ge \ell,$$

with the $r = 0$ condition $\Sigma_{1,1} \ge \ell$.

Per Definition 5.1, no coordinate is almost determined by the preceding coordinates, since each next coordinate retains nontrivial conditional variance. The condition rules out degenerate Gaussian directions that would make the density bound false.

For such a distribution, the following lemma says that no database vector is likely to lie near the challenge.

Lemma 5.2 (Gaussian proximity bound)**.** *Let $\mathcal{D}$ be an ℓ-variance-bounded multivariate normal distribution over $\mathbb{R}^d$. Then for all $p \in \mathbb{R}^d$ and $\varepsilon > 0$,*

$$\Pr_{q\sim\mathcal{D}}[\|p - q\|_\infty \le 2\varepsilon] \;<\; \left(1 - \frac{2}{\sqrt{2\pi}} \cdot \frac{2\varepsilon\sqrt{\ell}}{4\varepsilon^2 + \ell}\, e^{-\frac{4\varepsilon^2}{2\ell}}\right)^d.$$

In particular, for fixed $\varepsilon > 0$ and $\ell > 0$, with neither depending on d, the parenthesized quantity is a constant $c_{\varepsilon,\ell} \in (0,1)$, so the bound is $(c_{\varepsilon,\ell})^d$.

For our result, we augment the MIA game of Definition 4.4 against SAP, so that the adversary also receives the sampled scaling factor ρ as an input. We write $\mathbf{Adv}^{\mathsf{MIA},+\rho}_{\mathsf{SAP},\mathcal{D},n}$ for advantage in this game. We study experimentally whether the attack degrades when the scaling factor is the approximate one recovered by Getr rather than the exact ρ.

Theorem 5.3 (Attack with a known scaling factor)**.** *Let $\mathcal{D}$ be an ℓ-variance-bounded multivariate normal distribution over $\mathbb{R}^d$, suppose $\varepsilon > 0$ and $\beta \le 4\varepsilon$, and let $c_{\varepsilon,\ell}$ be the constant of Lemma 5.2. Then, for every $n \ge 1$ and $d \ge 1$, there is an adversary $\mathcal{A}^{\mathsf{MIA}*}_{\mathcal{D}}$ in the augmented game with*

$$\mathbf{Adv}^{\mathsf{MIA},+\rho}_{\mathsf{SAP},\mathcal{D},n}(\mathcal{A}^{\mathsf{MIA}*}_{\mathcal{D}}) \;>\; 1 - n\,(c_{\varepsilon,\ell})^d. \tag{2}$$

Proof. On input (H, p, ρ), the adversary $\mathcal{A}^{\mathsf{MIA}*}_{\mathcal{D}}$ runs the adversary $\mathcal{A}^{\mathsf{MIA}}_{\mathcal{D}}$ of Section 5.1 with the given ρ in place of the estimate, so that every hash value is divided by the exact scaling factor ρ. Let b' be its output. We bound its two error events and combine them.

When $b = 1$, the challenge vector p is in the database, so H contains its hash value $\rho p + \eta_p$ with $\|\eta_p\| \le \rho\beta/4$. The corresponding entry of the rescaled database $H' = \{h/\rho : h \in H\}$ is $p + \eta_p/\rho$, at distance $\|\eta_p\|/\rho \le \beta/4 \le \varepsilon$ from p. The adversary outputs 1, so $\Pr[b' = 1 \mid b = 1] = 1$.

When $b = 0$, the adversary returns 0 unless some database vector x_j gives a rescaled value within ε of p. The rescaled value for x_j is $x_j + \eta_j/\rho$, within $\beta/4 \le \varepsilon$ of x_j. By the triangle inequality, a rescaled value within ε of p requires $\|x_j - p\| \le 2\varepsilon$, and hence $\|x_j - p\|_\infty \le 2\varepsilon$. So the adversary returns 0 whenever no database vector lies within 2ε of p in ℓ_∞. By Lemma 5.2, each of the n database vectors is this close with probability less than $(c_{\varepsilon,\ell})^d$, so by a union bound

$$\Pr[\,\exists\, x_j \in S : \|x_j - p\|_\infty \le 2\varepsilon\,] \;<\; n(c_{\varepsilon,\ell})^d,$$

and $\Pr[b' = 0 \mid b = 0] > 1 - n(c_{\varepsilon,\ell})^d$.

Since the adversary always accepts in the member case and rejects with probability greater than $1 - n(c_{\varepsilon,\ell})^{\,d}$ in the nonmember case,

$$\begin{aligned}
&\mathbf{Adv}^{\mathsf{MIA},+\rho}_{\mathsf{SAP},\mathcal{D},n}(\mathcal{A}^{\mathsf{MIA}*}_{\mathcal{D}}) \\
&\quad = \Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0] \\
&\quad = \Pr[b' = 1 \mid b = 1] + \Pr[b' = 0 \mid b = 0] - 1 \\
&\quad > 1 - n(c_{\varepsilon,\ell})^{\,d},
\end{aligned}$$

which proves the claim. □

Proof of Lemma 5.2. For each coordinate i, let

$$C_i := \{|p[i] - q[i]| \le 2\varepsilon\}$$

be the event that coordinate i is close. Let $B_i := C_1 \cap \cdots \cap C_i$, and let B_0 be the whole sample space. Then the event $\|p - q\|_\infty \le 2\varepsilon$ is exactly B_d. By the chain rule,

$$\Pr_q[\|p - q\|_\infty \le 2\varepsilon] \;=\; \Pr[B_d] \;=\; \prod_{i=1}^{d} \Pr[C_i \mid B_{i-1}]. \tag{3}$$

We show that every factor in (3) is at most

$$\alpha := \Pr_{G\sim\mathcal{N}(0,1)}[\,|G| \le 2\varepsilon/\sqrt{\ell}\,].$$

Here G is a scalar standard normal random variable, so α is a one-dimensional Gaussian interval probability. Fix i and write $A_i := [\,p[i] - 2\varepsilon,\; p[i] + 2\varepsilon\,]$.

The event B_{i-1} does not fix the previous coordinates, but only says that all previous coordinate tests hold; equivalently, $q[:i{-}1]$ lies in the Cartesian product

$$\prod_{j=1}^{i-1} [\,p[j] - 2\varepsilon,\; p[j] + 2\varepsilon\,].$$

Thus $\Pr[C_i \mid B_{i-1}]$ is obtained by first choosing the random prefix $q[:i{-}1]$ conditioned on B_{i-1}, and then asking whether coordinate i is close. Equivalently,

$$\Pr[C_i \mid B_{i-1}] = \mathbb{E}_{v\sim(q[:i-1]|B_{i-1})}\left[\Pr[q[i] \in A_i \mid q[:i{-}1] = v]\right]. \tag{4}$$

So it is enough to show that every term inside the expectation in (4) is at most α.

Fix a prefix vector $v = (v_1, \ldots, v_{i-1})$, with v empty when $i = 1$. The conditional distribution of the next coordinate is Gaussian:

$$q[i] \mid q[:i{-}1] = v \sim \mathcal{N}(\mu_i(v), \sigma_i^2).$$

For a jointly Gaussian vector, once we fix the previous coordinates, the conditional variance of the next coordinate is determined by the covariance matrix and is therefore the same for every choice of v; only the conditional mean $\mu_i(v)$ changes. The ℓ-variance-bounded assumption says that this common variance is at least ℓ.

The interval A_i has length 4ε, and a scalar Gaussian assigns the largest probability to an interval of this length when it is centered at the mean. Since the conditional Gaussian above has mean $\mu_i(v)$ and standard deviation σ_i, with $\sigma_i^2 \geq \ell$, we obtain

$$\begin{aligned}
&\Pr_{Z\sim\mathcal{N}(\mu_i(v),\sigma_i^2)}[Z \in A_i] \\
&\quad \leq \Pr_{G\sim\mathcal{N}(0,1)}[\,|G| \leq 2\varepsilon/\sigma_i\,] \\
&\quad \leq \Pr_{G\sim\mathcal{N}(0,1)}[\,|G| \leq 2\varepsilon/\sqrt{\ell}\,] = \alpha.
\end{aligned}$$

Thus every term in the expectation in (4) is at most α, so

$$\Pr[C_i \mid B_{i-1}] \leq \alpha \qquad \text{for every } i = 1, \ldots, d.$$

For completeness, let $\varphi(t) = e^{-t^2/2}/\sqrt{2\pi}$ and $I(t) = \Pr[Z > t] = \int_t^\infty \varphi(x)\,dx$ for a scalar standard normal random variable Z. Since $\varphi'(x) = -x\varphi(x)$, integration by parts gives

$$I(t) = \frac{\varphi(t)}{t} - \int_t^\infty \frac{\varphi(x)}{x^2}\,dx > \frac{\varphi(t)}{t} - \frac{I(t)}{t^2}.$$

Rearranging gives $I(t) > \varphi(t)t/(t^2+1)$. Writing $\Pr[|Z| \leq t] = 1 - 2I(t)$ and substituting $t = 2\varepsilon/\sqrt{\ell}$ gives

$$\Pr\big[|Z| \leq 2\varepsilon/\sqrt{\ell}\big] \;<\; 1 \;-\; \frac{2}{\sqrt{2\pi}} \cdot \frac{2\varepsilon\sqrt{\ell}}{4\varepsilon^2 + \ell}\, e^{-2\varepsilon^2/\ell}. \tag{5}$$

Let $c_{\varepsilon,\ell}$ denote the right-hand side of (5), a number in $(0,1)$ depending only on ε and ℓ. Since every one of the d factors in (3) is less than $c_{\varepsilon,\ell}$, one for each coordinate,

$$\Pr_q[\|p - q\|_\infty \leq 2\varepsilon] < (c_{\varepsilon,\ell})^{\,d},$$

which proves both the stated bound and the "in particular" claim. □

Interpreting the SAP bound. Set

$$t = \frac{2\varepsilon}{\sqrt{\ell}}, \qquad \varphi(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2}, \qquad a = \frac{2\varphi(t)t}{t^2+1}.$$

Then $c_{\varepsilon,\ell} = 1 - a$ and

$$n(c_{\varepsilon,\ell})^{\,d} = n(1-a)^d \leq ne^{-ad}.$$

Thus, $d \geq \ln(n/\nu)/a$ suffices for advantage greater than $1 - \nu$. At the defaults $\varepsilon = 0.5$, $\ell = 1$, and $n = 2{,}000$, the error term $n(c_{\varepsilon,\ell})^{\,d}$ is below 0.002 at $d = 50$. The bound is loose as it is vacuous at $d = 10$ and first becomes positive at $d = 28$, although the measured advantage is already near one at $d = 10$. The looseness does not come from the union bound. It comes mainly from replacing the Euclidean condition $\|x_j - p\| \leq 2\varepsilon$ by the much weaker condition $\|x_j - p\|_\infty \leq 2\varepsilon$, and from bounding each coordinate for the worst-case challenge.

Example and limitations. As a concrete example, take unit-variance coordinates ($\ell = 1$) and dimension $d = 384$. With $\varepsilon = 0.25$ we have $c_{\varepsilon,\ell} \approx 0.72$, so even at a billion database vectors ($n = 10^9$) the bound gives advantage greater than $1 - 10^{-46}$. A tighter threshold $\varepsilon = 0.1$ gives $c_{\varepsilon,\ell} \approx 0.85$, and advantage greater than $1 - 10^{-18}$ at the same size.

Also, observe that Theorem 5.3 is stated and proved for ℓ-variance-bounded multivariate normal distributions, but neither constraint holds on the real embeddings we attack, which are normalized to the unit sphere rather than Gaussian with full support. We thus experimentally validate the attack on real embeddings in Section 5.2.

Instantiating Getr. It remains to give the subroutine Getr. Recall that $\mathsf{H}_\rho(x)$ outputs $\rho x + \eta$, where η is uniform on the ℓ_2 ball of radius $\rho\beta/4$ and is sampled fresh and independently of x at each evaluation. Hashing multiplies each coordinate by ρ and adds independent noise whose own per-coordinate variance is proportional to ρ^2. This gives the following exact relation between the average hash-value variance and $\bar{\sigma}_x^2$, which we then invert.

Proposition 5.4 (Identity for the scaling factor)**.** *Let $h = \rho x + \eta$, where $x \leftarrow\!\!\$\ \mathcal{D}$ and the noise η is uniform on the ℓ_2 ball of radius $\rho\beta/4$ in $\mathbb{R}^d$ and independent of x. Let $\bar{v}_h = \frac{1}{d}\sum_{i=1}^{d} \mathrm{Var}(h[i])$ and $\bar{\sigma}_x^2 = \frac{1}{d}\sum_{i=1}^{d} \mathrm{Var}(x[i])$, assuming these variances are finite. Then*

$$\bar{v}_h \;=\; \rho^2\left(\bar{\sigma}_x^2 + \frac{\beta^2}{16(d+2)}\right).$$

Proof. Fix a coordinate i. The noise η is independent of x and has mean zero, so

$$\mathrm{Var}(h[i]) = \rho^2\,\mathrm{Var}(x[i]) + \mathrm{Var}(\eta[i]).$$

The noise η is uniform on the ℓ_2 ball of radius $R = \rho\beta/4$ in $\mathbb{R}^d$, so $\mathbb{E}[\|\eta\|^2] = R^2 d/(d+2)$, and by symmetry every coordinate has the same variance $\mathrm{Var}(\eta[i]) = R^2/(d+2) = \rho^2\beta^2/(16(d+2))$. Averaging $\mathrm{Var}(h[i])$ over the d coordinates gives the stated identity. □

Whenever $\bar{\sigma}_x^2 + \beta^2/(16(d+2)) > 0$, this factor is a constant known to the adversary, since $\bar{\sigma}_x^2$ comes from $\mathcal{D}$ and both β and d are public. The identity then determines ρ, and inverting it defines Getr for $n \geq 2$, as follows:

$\mathsf{Getr}(\bar{\sigma}_x^2,\ \beta,\ H)$	
Input: the hashed database $H = (h_1, \ldots, h_n)$, each $h_j \in \mathbb{R}^d$.	
$c_\beta \leftarrow \bar{\sigma}_x^2 + \beta^2/(16(d+2))$	
For $i = 1, \ldots, d$:	
$\quad \hat{\mu}_i \leftarrow \frac{1}{n}\sum_{j=1}^{n} h_j[i]$	// mean of coordinate i
$\quad \hat{v}_i \leftarrow \frac{1}{n-1}\sum_{j=1}^{n}\big(h_j[i] - \hat{\mu}_i\big)^2$	// sample variance
$\hat{v}_h \leftarrow \frac{1}{d}\sum_{i=1}^{d} \hat{v}_i$	// average over coordinates
Return $\hat{\rho} \leftarrow \sqrt{\hat{v}_h / c_\beta}$	
$\quad$ // positive root; $\rho > 0$ by Kg	

In Getr, only the empirical average variance $\hat{v}_h$ depends on the hashed database; the denominator is fixed by $\mathcal{D}$ and the public parameter β. For independent samples with finite variance, the law of large numbers makes $\hat{v}_h$ approach $\bar{v}_h$ as the database grows. The identity for the scaling factor then makes $\hat{\rho}$ approach ρ. Note that the attack is unaffected by shuffling the database, since $\hat{v}_h$ is invariant under permutations of H. The attack also works equally well for every fixed key ρ, because $\hat{\rho}/\rho$ and the rescaled values $h/\hat{\rho}$ have the same distribution for every value of ρ.

Using an estimated scaling factor. Theorem 5.3 gives the adversary the secret ρ, whereas the implemented attack uses $\hat{\rho}$. For $y = h/\rho$ and $\hat{y} = h/\hat{\rho}$,

$$\|\hat{y} - y\| = \left|\frac{\rho}{\hat{\rho}} - 1\right| \|y\|.$$

Let E be the event

$$\left|\frac{\rho}{\hat{\rho}} - 1\right| \max_{h \in H} \|h/\rho\| \leq \varepsilon - \beta/4. \tag{6}$$

Since $\|h/\rho - x\| \le \beta/4$, on E the member hash remains within ε of the challenge. In the nonmember case, acceptance and the triangle inequality imply that the corresponding plaintext lies within 2ε of the challenge, exactly as in Theorem 5.3. Hence, if $\Pr[\overline{E} \mid b] \le \zeta$ for both bits, then

$$\mathbf{Adv}^{\mathsf{MIA}}_{\mathsf{SAP},\mathcal{D},n}(\mathcal{A}^{\mathsf{MIA}}_{\mathcal{D}}) > 1 - n(c_{\varepsilon,\ell})^{\,d} - 2\zeta.$$

Bounding ζ requires showing that, on a database of finite size, Getr returns a value close to ρ and the largest hash norm is not much larger than typical; we leave this open and test the implemented attack experimentally.

Estimating the population variance. The game gives the adversary $\bar{\sigma}_x^2$ exactly, but an attacker in practice must estimate it from public data. Suppose the estimate has relative error δ and write $c = \beta^2/(16(d+2))$. Replacing $\bar{\sigma}_x^2$ by $(1+\delta)\bar{\sigma}_x^2$ multiplies $\hat{\rho}$ by

$$\left(1 + \delta \frac{\bar{\sigma}_x^2}{\bar{\sigma}_x^2 + c}\right)^{-1/2},$$

so the exact induced relative change is the absolute difference of this factor from one (approximately $|\delta|/2$ for small δ when $c \ll \bar{\sigma}_x^2$). At the MS-MARCO defaults ($\varepsilon = 0.5$, $\beta = 1$), condition (6) still holds, counting only this source of error, for an overestimate of up to about 44% or an underestimate of up to about 36%. This is not an end-to-end finite-sample robustness guarantee; the required bound on ζ remains open.

5.2 Experimental Evaluation

We test the attack on synthetic Gaussian vectors and MS-MARCO passage embeddings. The adversary has no known plaintexts ($k = 0$), and we vary the database size and the noise parameter, so any advantage we experimentally measure comes from public information about the distribution alone. The synthetic experiments match the distributional model used in our analysis of SAP. The MS-MARCO experiments test whether the same attack works on normalized vectors produced by a standard embedding model.

Figure 3 shows that the attack remains strong when the scaling factor is estimated by Getr from the hashed database rather than given exactly. At the synthetic baseline, the advantage is 0.999; it remains 0.969 even at $d = 5$, reaches at least 0.999 for every tested $d \ge 10$, and changes little as we vary the database size. These results are consistent with Theorem 5.3, although that theorem analyzes the idealized test that knows the scaling factor exactly. On MS-MARCO, at the baseline, advantage is 0.997 with standard error 0.0013, and it remains at least 0.991 over database sizes from 500 to 10,000 with every calibration and held-out run drawing its own key. There the attack accepts every member and 0.003 of nonmembers. The held-out set has equal numbers of members and nonmembers, so almost everything the attack accepts is a member. Advantage is 0.989 at $\beta = 2$, the largest noise we test, so raising β does not defend SAP. Since every MS-MARCO vector in this experiment has norm one, the result also shows, for this dataset and model, that the attack does not rely on either a Gaussian input distribution or variation in vector norms. The setup and parameters are in Appendix F.1.

Recovering the scaling factor. Figure 4 shows the accuracy of recovering the scaling factor on its own, apart from the membership test. At the default setting, the relative error of Getr is 0.351% on the synthetic data and 0.0287% on MS-MARCO, never exceeding 0.063% across the tested MS-MARCO database sizes. On MS-MARCO it falls from 0.0287% at $n = 2{,}000$ to 0.0126%

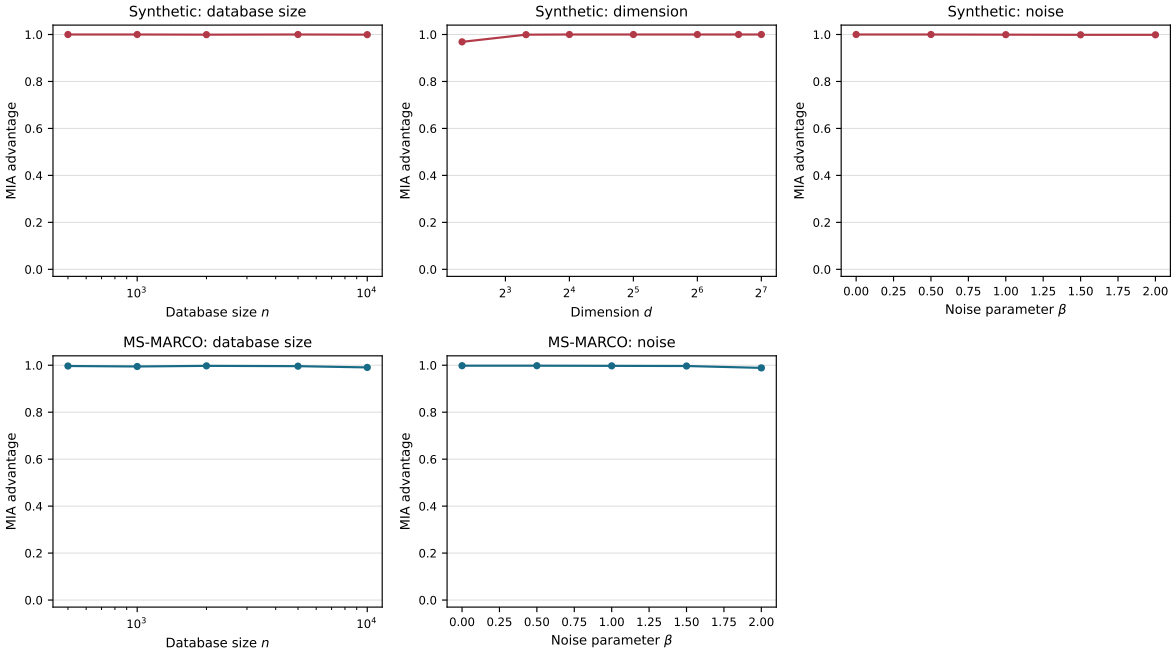

Figure 3: Membership-inference advantage against SAP with a threshold chosen on separate calibration runs.

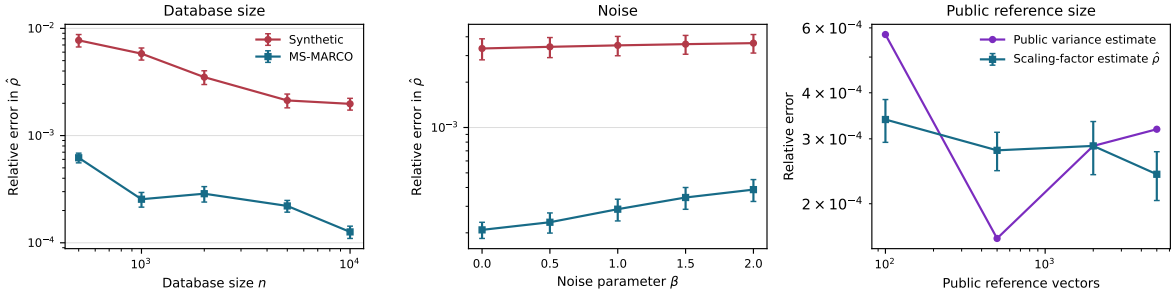

Figure 4: Relative error in recovering the secret scaling factor. The first two panels vary the database size and noise. The last panel varies the size of the public MS-MARCO reference set and shows both the error in its variance estimate and the resulting error in the scaling-factor estimate. Error bars show one standard error. The reference-size panel uses one public reference sample per size, so its error bars do not include the variation from resampling that reference.

at $n = 10{,}000$. Five times the data gives about 2.3 times less error, which is compatible with $1/\sqrt{nd}$ behavior but does not establish a scaling law. We make no claim beyond the tested sizes. As the public reference set ranges from 100 to 5,000 vectors, the variance error relative to the full 24,000-vector development pool remains below 0.058%, while the resulting scaling-factor error remains below 0.034%. Each size uses one reference draw, so this experiment does not measure variability over reference resampling. In this tested setting, neither normalization nor replacing a challenge-dependent variance estimate with a disjoint public estimate prevents recovery of the scaling factor.

6 Rotate-and-Perturb and its Security

We introduce an alternative to SAP, which we call *Rotate-and-Perturb* (RAP), aimed at membership privacy on unit vectors. It maps x as $Mx + \eta$, where M is a secret orthogonal matrix, rather than a secret scaling factor as in SAP, and η is fresh bounded noise. We remind the reader that we treat keyed hashing schemes as secret-key and randomized.

Relation to prior and concurrent work. FGHO [FGHO22] define a generalization of SAP they call SAP-I, which multiplies the vector by a secret orthogonal matrix before scaling it, adding a secret translation, and adding noise, and they leave its security unanalyzed. RAP is SAP-I with the scaling factor set to 1, no translation, and the inputs restricted to the unit sphere. We remove the scaling factor because retaining it would reintroduce the database-wide statistic that our SAP attack exploits, the average coordinate variance of Section 5.1. A common translation does not change pairwise distances, so it provides no protection against the geometric attacks studied here and only adds notation.

Concurrent unpublished work by Jutla, Kirat, Manohar, and Patranabis [JKMP26], shared with us by the authors, studies the same variant and prompted our unmatched attacks. They consider known plaintexts without known plaintext–ciphertext correspondences and formulate recovery as a labeled subgraph-isomorphism problem.

6.1 The Scheme

Fix a public noise parameter $\beta \geq 0$ and sample the key as a uniformly random orthogonal matrix $M \leftarrow\!\!\$\ O(d)$. On input $x \in \mathbb{R}^d$, the hash samples fresh noise η uniformly from the Euclidean ball of radius $\beta/4$ and returns

$$\mathsf{H}_M(x) = Mx + \eta.$$

Here and below, "rotation" means an orthogonal transformation, which may include a reflection. Setting $\beta = 0$ gives a noiseless secret rotation and already suffices for the basic security result without known plaintexts below. We nevertheless include noise because it makes the inner products exploited by our known-plaintext attacks only approximate, weakening the attacks in both the matched and unmatched settings, with noise providing the only protection in the former.

On vector normalization. Although hashing is defined on all of $\mathbb{R}^d$, RAP does not hide the input norm. Indeed, if $h = Mx + \eta$, then

$$\big|\|h\| - \|x\|\big| = \big|\|Mx + \eta\| - \|Mx\|\big| \leq \|\eta\| \leq \beta/4.$$

The noiseless scheme therefore reveals the norm exactly, and the noisy scheme reveals it within $\beta/4$. This is why our proposed use of RAP and our main positive result restrict the database vectors

to the unit sphere. On unit inputs the remaining departure from the sphere says nothing about the database, because $\|Mx + \eta\|$ has the same distribution for every unit vector x. Many cosine-similarity deployments normalize already; the restriction excludes retrievers and recommenders that rank by unnormalized inner product [KOM+20, GSL+20].

Correctness. For independently generated hash values $h_x = Mx + \eta_x$, $h_y = My + \eta_y$, and $h_z = Mz + \eta_z$, orthogonality of M and the triangle inequality give $\big|\,\mathsf{dist}(h_x, h_y) - \mathsf{dist}(x, y)\,\big| \le \|\eta_x - \eta_y\| \le \beta/2$. A distance comparison involves two such distances, so it can be off by at most β: if $\mathsf{dist}(x, y) < \mathsf{dist}(x, z) - \beta$, then $\mathsf{dist}(h_x, h_y) < \mathsf{dist}(h_x, h_z)$ for every choice of the hashing randomness, and hence RAP is a β-DCPH scheme.

6.2 RAP through Noisy Gram Leakage

Here we give a leakage profile equivalent to RAP's trivial leakage profile as previously defined, which is convenient for the analysis that follows. Namely, recall from Definition 4.3 that every DCPH scheme has a trivial leakage profile; for RAP, we denote it by $\mathcal{L}^{\mathrm{triv}}_{\mathrm{RAP},\beta}$. On input $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$, it samples $M \leftarrow\!\!\$\ O(d)$ and $\eta_1, \ldots, \eta_n \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4)$, independent of each other and of M, and outputs

$$\mathcal{L}^{\mathrm{triv}}_{\mathrm{RAP},\beta}(S) = (Mx_1 + \eta_1, \ldots, Mx_n + \eta_n).$$

We also define the noisy Gram leakage profile.

Definition 6.1 (Noisy Gram leakage profile)**.** Fix $\beta \ge 0$ and let $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$. Sample $\eta'_1, \ldots, \eta'_n \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4)$ independently and set

$$\mathcal{L}_{\mathrm{Gram},\beta}(S) = \Gamma(x_1 + \eta'_1, \ldots, x_n + \eta'_n),$$

where Γ is the Gram matrix from (1).

Lemma 6.2 (RAP leakage profile)**.** *The leakage profiles $\mathcal{L}^{\mathrm{triv}}_{\mathrm{RAP},\beta}$ and $\mathcal{L}_{\mathrm{Gram},\beta}$ are equivalent.*

The proof is in Appendix A.

6.3 Security for Uniform Inputs on the Sphere

We first consider the basic game of Section 4, where the adversary receives only the hashed challenge database and the unhashed challenge vector. For inputs sampled uniformly from the unit sphere in $\mathbb{R}^d$, the following result is exact and continues to hold when $\beta = 0$, although we retain noise for the known-plaintext attack setting considered later.

Theorem 6.3 (Perfect security under uniform inputs)**.** *Let $\mathcal{D} = \mathrm{Unif}(S^{d-1})$. Then, for every $\beta \ge 0$, every $n \ge 1$, and every adversary $\mathcal{A}$,*

$$\mathbf{Adv}^{\mathsf{MIA}}_{\mathsf{RAP},\mathcal{D},n}(\mathcal{A}) = 0.$$

We use one elementary observation. If a random tuple $(X_1, \ldots, X_n)$ has a distribution unchanged by every common orthogonal transformation, and M is an independent uniform orthogonal matrix, then

$$(MX_1, \ldots, MX_n) \equiv (X_1, \ldots, X_n).$$

Indeed, conditional on $M = U$, the left-hand side has the distribution of $(UX_1, \ldots, UX_n)$, which is the original distribution for every fixed U.

Proof of Theorem 6.3. Let Q_β be the distribution of $X + \eta$ for $X \leftarrow\!\!\$\ \mathrm{Unif}(S^{d-1})$ and independent $\eta \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4)$.

When the challenge is not in the database, the challenge p is independent of all database vectors. By the observation above, applying the common secret rotation does not change the joint distribution of the database vectors. After adding the independent noise, the n hash values are independent samples from Q_β and are independent of p.

Now suppose the challenge is planted, and condition on its planted position. It is enough to consider the first position. Condition further on a fixed challenge value $p = p_0$. The vector Mp_0 is uniform on the sphere. Given M, every other rotated database vector MX_i is also uniform on the sphere, and their joint conditional distribution does not depend on M. Consequently they are independent of Mp_0. After adding the independent noise, all n hash values are again independent samples from Q_β, independent of the fixed value p_0. Averaging over p_0 gives exactly the same view as in the nonmember case. The two views are therefore identically distributed, so every adversary has advantage zero. □

6.4 Attacks Using Non-Uniformity

Real embeddings are spread far more along some directions than others, and which directions those are is a property of the embedding model, so an adversary can estimate them from public data. RAP applies one secret rotation to every vector, which moves those directions without changing the pattern they form. An adversary that could find them in the hash values and compare them with the ones it knows from public data could recover the rotation. We therefore ask whether this non-uniformity alone breaks RAP with no known plaintexts.

Attacks. Both attacks we tried aim to recover M. In prior work, Giannella et al. [GLK13] recover an orthogonal transform from known input–output pairs, but no such pairs are available here, so both attacks use public distribution information instead. The first estimates the directions of largest spread of $\mathcal{D}$ from a public reference sample, estimates the corresponding directions of the hashed database, and aligns the two. The second alternates between matching each hash value to a nearby plaintext and solving for the orthogonal matrix that best fits the current matching, which is the orthogonal Procrustes problem and has a closed-form solution. We find that both fail on a challenge database of $n = 2{,}000$ MS-MARCO vectors, given a public reference sample of 6,000 further vectors from the same corpus, disjoint from the database, which is what the attacks use to estimate the statistics of $\mathcal{D}$. At $\beta = 0.5$ the first attack reaches AUC 0.434 with 95% bootstrap interval $[0.365, 0.504]$, and the second reaches 0.471 $[0.398, 0.548]$ when its initial rotation is the one found by the first attack and 0.492 $[0.423, 0.561]$ when it is a random rotation. Every interval contains 1/2.

Control. As a control, we ran the scoring step with the information the attacks must otherwise estimate. Given the true rotation it separates members from nonmembers at AUC 0.993 with 95% bootstrap interval $[0.978, 1.000]$, and given the true matching the Procrustes step of the second attack recovers M and reaches AUC 1.000. What stops the second attack is therefore guessing the correspondence between hash values and reference vectors, rather than anything structural about RAP. For the first attack, enlarging the public reference sample from 500 to 6,000 vectors does not help, with AUC 0.512, 0.507, and 0.455, so its failure is not a matter of reference sample size.

We also gave the second attack the k known plaintexts of the matched setting, seeding its alternation with the known matchings and holding them fixed at every round. On MS-MARCO at $\beta = 0.5$ its advantage at $k = 3$ is 0.026 ± 0.028 at $n = 100$ and 0.029 ± 0.012 at $n = 500$, and

it does not reach 0.5 through the fifty known plaintexts we test, rising only to 0.351 ± 0.023 and 0.471 ± 0.023 there. It is therefore far weaker than the attack of Section 6.5, which reaches 0.5 at $k = 3$.

6.5 Matched Attack

We study the matched setting of Definition 4.5 against the noisy Gram leakage profile $\mathcal{L}_{\mathrm{Gram},\beta}$ of Definition 6.1, which by Lemma 6.2 is equivalent to RAP's own leakage. The adversary sees the matrix G that this profile outputs, and its entry $G[a, b]$ is the inner product of the noisy versions of the two database vectors at positions a and b. Since the database vectors are unit vectors, both our attacks use the normalized entries $\widehat{G}[a, b] = G[a, b]/\sqrt{G[a, a]\, G[b, b]}$, which the adversary can compute from G alone. Each such entry is within β of $\langle x_a, x_b \rangle$ (Appendix B.1).

The attack. First, relabel so that the known positions are the first k, so the adversary knows $x_1, \ldots, x_k$ along with the challenge vector p in the clear. For each candidate position $j > k$ for p, the attack computes the root-mean-square difference, over the k known plaintexts, between the inner product with p and the normalized entry $\widehat{G}[i, j]$, and returns *member* if the best-fitting position's mismatch falls below a threshold t chosen on calibration runs.

Matched membership attack

Input: known plaintexts $x_1, \ldots, x_k$, at positions $1, \ldots, k$ of the challenge database; challenge vector p; noisy Gram matrix G of the challenge database; threshold t.

1. For each position $j = k + 1, \ldots, n$, set $s(j) \leftarrow \left(\frac{1}{k} \sum_{i=1}^{k} \big(\langle x_i, p \rangle - \widehat{G}[i, j] \big)^2 \right)^{1/2}$.
2. Set $s \leftarrow \min_{j>k} s(j)$.
3. Return 1 if $s \le t$, and return 0 otherwise.

In step 1, $s(j)$ is the root-mean-square difference, over the k known plaintexts, between the inner product with p and the normalized entry for position j, and in step 2, s is the mismatch of the best-fitting position. The threshold t is chosen on calibration runs.

This test works because if p is in the database, at some position $j > k$ unknown to the adversary, then for every known plaintext x_i the adversary can compute $\langle x_i, p \rangle$ from what it holds: the leakage gives it $\widehat{G}[i, j]$, and the two agree up to the noise, so the mismatch is small. If instead p is not in the database, then no position agrees on all k of these quantities except by chance.

Experiments. In our experiments, we choose the threshold t on separate calibration runs. Production databases are far larger than the sizes we can sweep, so we report how the breakpoint grows with n rather than its value at one size. On MS-MARCO at $\beta = 0.5$ the breakpoint is

$$\begin{aligned} k &= 3, \quad 4, \quad\quad 4, \quad 5, \quad\quad 5, \quad 7, \quad \text{for} \\ n &= 10^2,\ 5 \cdot 10^2,\ 10^3,\ 2 \cdot 10^3,\ 10^4,\ 10^5, \text{ resp.} \end{aligned}$$

At these breakpoints the attack accepts between 0.81 and 0.92 of members and between 0.13 and 0.33 of nonmembers. Three orders of magnitude in database size raise the breakpoint by four known plaintexts, or about one per two to three doublings. The acceptance rates at the breakpoints and the effect of raising the noise to $\beta = 1$ are in Appendix B.2, and Appendix F.2 has the effect of letting the adversary choose its known plaintexts. The unmatched attack of Section 6.6 needs five known plaintexts at $n = 100$, so at that size knowing the matching lowers the breakpoint by about

two. The gap widens as the database grows, since the unmatched attack must also locate its known plaintexts among more candidates.

The attack's efficacy is not specific to text embeddings. Repeating the $n = 100$, $\beta = 0.5$ case on CLIP embeddings of 10,000 MS COCO images gives the same breakpoint: $k = 3$. We also run the matched attack on 512-dimensional FaceNet [SKP15] embeddings of the funneled LFW face images [HRBLM07], where the breakpoint is $k = 2$ at $n = 100$ and $k = 3$ at $n = 1{,}000$ among the values of k we tested. Together with the MS-MARCO and CLIP/MS COCO results, this shows the same attack working on text, multimodal, and face embedding pipelines. The LFW setup, acceptance rates, and caveats about FaceNet's pretraining data are in Appendix B.2.

6.6 Unmatched Attack

We study security of RAP according to Definition 4.5 with $\mathcal{L}_{\mathrm{Gram},\beta}$ and $r = \mathsf{unmatch}$. Shuffling and noise create different obstacles here. Shuffling hides the mapping between known plaintexts and positions, while noise makes the relevant inner products only approximate. Without noise, one known plaintext x_1 already allows an exact membership test. If p is a member, then $\langle x_1, p\rangle$ appears in the Gram matrix. An independent and random nonmember satisfies this equality with probability zero in theory, and with low probability in an implementation, where finite precision may make distinct inner products collide. Noise prevents this exact test and reduces the advantage of this approach at a fixed k.

The attack. Fix $k \geq 2$ and $\beta < 4$, the latter so that every vector $x + \eta$ is nonzero. The attack uses the normalized entries $\widehat{G}$ of Section 6.5. It scores an assignment of the known plaintexts x_i to positions by how well their known pairwise inner products match the corresponding entries of $\widehat{G}$, and adds the same comparison of each x_i with the challenge p at an unassigned position. It minimizes that score approximately by a beam search [Low76] of width B and thresholds the result on calibration runs. For $2 \leq a \leq k$, a partial assignment π maps $x_1, \ldots, x_a$ to distinct positions. Its error is

$$E_a(\pi) = \sum_{1 \leq r < s \leq a} \big(\langle x_r, x_s\rangle - \widehat{G}[\pi(r), \pi(s)]\big)^2.$$

For a complete assignment π and an unassigned position j, define the joint error

$$T_p(\pi, j) = E_k(\pi) + \sum_{a=1}^{k} \big(\langle x_a, p\rangle - \widehat{G}[\pi(a), j]\big)^2.$$

The attack minimizes $T_p(\pi, j)$ approximately, by a beam search [Low76] of width B.

Beam-search unmatched membership attack

Input: known plaintexts $x_1, \ldots, x_k$; challenge vector p; noisy Gram matrix G of the challenge database; beam width $B \geq 1$; threshold t

1. Set $\widehat{G}[i, j] \leftarrow G[i, j]/\sqrt{G[i, i]\, G[j, j]}$ for all $i, j \in [n]$.
2. Reorder and relabel $x_1, \ldots, x_k$ by the ordering rule below.
3. Let Π_2 be the $\min\{B, n(n-1)\}$ one-to-one maps $\pi : \{1, 2\} \to [n]$ with the smallest $E_2(\pi)$.
4. For $a = 3, \ldots, k$, let Π_a be the at most B one-to-one maps $\pi : \{1, \ldots, a\} \to [n]$ extending a map in Π_{a-1}, with the smallest $E_a(\pi)$.
5. Set $\widehat{s}_B(p) \leftarrow \sqrt{\dfrac{2}{k(k+1)} \min_{\substack{\pi \in \Pi_k \\ j \notin \pi(\{1,\ldots,k\})}} T_p(\pi, j)}$.
6. Return 1 if $\widehat{s}_B(p) \leq t$, and 0 otherwise.

Here $E_a(\pi)$ measures how well the assigned positions reproduce all pairwise inner products among the first a known plaintexts, and the second term of $T_p(\pi, j)$ tests whether position j has the inner products expected of the challenge. Exhaustively minimizing T_p would require considering far too many assignments of the known plaintexts to positions, so we view the search as a tree. A node at depth a assigns the first a known plaintexts to distinct positions, and its children try every unassigned position for the next plaintext. An exhaustive search would retain every node and have about n^k leaves, whereas beam search retains only the B maps with the smallest error at each depth, extends those, and discards the rest. This makes the attack efficient, but it is only heuristic, and once the actual assignment falls outside the beam, later steps cannot recover it.

The threshold t is chosen on calibration runs. The ordering in step 2 places the most unusual inner-product constraints first, in the sense made precise below, before the beam is pruned, and uses only the known plaintexts and the leakage. It starts with the pair of known plaintexts whose inner product is farthest from its tenth nearest entry in $\widehat{G}$. A pair whose inner product lies far from the entries of $\widehat{G}$ fits few candidate position pairs, so applying it first discards the most assignments while the beam is still wide. We measure that rarity against the tenth nearest entry rather than the nearest so that a single nearby entry does not decide the ordering. The rule then repeatedly appends the known plaintext whose inner products with those already ordered deviate most from the mean entry of $\widehat{G}$, in units of the standard deviation of those entries. Ties in the ordering or in the beam may be broken arbitrarily.

Normalizing G takes $O(n^2)$ arithmetic operations, the inner products among the known plaintexts and with the challenge take $O(k^2 d)$, the ordering rule takes $O(k^2 n^2)$, and the beam search takes $O(n^2 + Bnk^2)$, so the attack runs in polynomial time.

Experiments. We evaluate on the same 384-dimensional normalized MS-MARCO embeddings used above and on vectors uniform on the unit sphere, with thresholds chosen on separate calibration runs. The advantage rises with k: on MS-MARCO at $n = 100$ it is 0.21, 0.60 and 0.95 for $k = 4, 5, 6$, measured with a fresh key drawn for every run, with the same measurements in Table 1; at $n = 50$, in the sweep shown in Figure 6, it is 0.48, 0.91 and 0.97. At the $n = 100$ breakpoint the attack accepts 0.87 of members and 0.28 of nonmembers. For data uniformly random on the sphere, we show the same pattern with slightly more known plaintexts, so the attack is not specific to the distribution of MS-MARCO embeddings. The acceptance rates, noise and database-size sweeps, search-budget effects, and relation to recovering the matching are in Appendices B.4 and B.5.

6.7 Known-Plaintext Security for Uniform Inputs on the Sphere

The attacks of the two preceding subsections both need a number of known plaintexts that grows about logarithmically in the database size. Here, we prove corresponding upper bounds on the advantage of any adversary for data uniform on the sphere. We work in the matched setting, which also covers the unmatched setting, since a matched adversary can ignore the matching it is given.

The proof reduces the matched known-plaintext attack to the *planted sketch problem*, a distinguishing problem we introduce. A *sketch* of a vector consists of the first k coordinates and the norm of the vector plus fresh noise. The distinguisher sees N sketches of unit vectors, one of which may be a sketch of a challenge vector it knows, and it must decide whether that planted sketch is present. The reduction rotates the subspace spanned by the k known plaintexts onto the first k coordinates, after which each unknown row (a noisy database vector) reveals just those k coordinates and its norm, and the planted row is hidden among $N = n - k$ candidates.

Theorem 6.4 (Matched known-plaintext security of RAP)**.** *There are absolute constants $c, C > 0$ such that the following holds. Let $\mathcal{D} = \mathrm{Unif}(S^{d-1})$, let $1 \le k < \min\{n, d\}$, and let $\beta \in (0, 2]$. Then every matched known-plaintext adversary $\mathcal{A}$ against* RAP *satisfies*

$$\mathbf{Adv}^{\mathsf{MIA}^{\mathsf{match}}}_{\mathsf{RAP},\mathcal{D},n,k}(\mathcal{A}) \le \frac{C}{\sqrt{n-k}} \left(\frac{c}{\beta}\right)^{k/2}.$$

The bound improves as the database grows and degrades as the adversary learns more plaintexts. We define the planted sketch problem and state the two results used in the proof below. Their proofs are in Appendix A.

A numerical illustration of the bound. For data uniform on the sphere, on a database of a million vectors, against an adversary holding three matched known plaintexts, we estimate the adversary's advantage is at most 0.019 when $d = 384$ and $\beta = 0.5$, the parameters of our MS-MARCO experiments. The theorem's constants are unspecified, but to obtain estimates, Appendix A.5 instead computes by numerical integration the exact quantity the theorem bounds, and gives the other database sizes and values of k. That bound grows by about a factor of 3.36 for each further known plaintext and shrinks by a factor of $\sqrt{10} \approx 3.16$ when the database grows tenfold. So the number of known plaintexts below which the bound certifies advantage under 0.5 grows by about one for every tenfold increase in the database size. This describes the bound, not any attack.

6.8 Proof Roadmap for the Planted Sketch Problem

The argument has two steps.

1. We reduce the matched known-plaintext attack to a standalone problem, the *planted sketch problem* (Theorem 6.5).

   The reduction receives only the challenge and the sketches and must build the adversary's view: the noisy Gram matrix, the known plaintext–position pairs, and the challenge. It samples the known plaintexts itself and uses an orthogonal change of coordinates to put their span into the fixed subspace $E = \mathrm{span}(e_1, \ldots, e_k)$ spanned by the first k standard basis vectors. This is only a choice of coordinates: Gram matrices are unchanged by applying the same orthogonal map to every row. Each sketch then fixes the projection of one unknown row onto E and fixes that row's norm. The only missing information is the direction of the remaining component in $E^\perp$. For ordinary rows that direction is uniform once the sketch is fixed. The planted row is the subtle case, because its hidden direction can remain correlated with the challenge. Lemma A.2 shows why one common random rotation of $E^\perp$ can randomize this hidden direction without changing any entry of the Gram matrix.

2. We then bound the advantage of every distinguisher against the planted sketch problem. We measure how far apart the planted and unplanted views are by their *chi-square divergence*; the advantage of every distinguisher is at most half the square root. The planted sketch lies at one of the $N = n - k$ positions, chosen uniformly at random. This uncertainty makes that divergence exactly $1/N$ times the divergence for a single sketch (Lemma A.3). For a single sketch, the noisy norm has the same distribution in either case, because every database vector is a unit vector, so the distinguisher can only exploit the k revealed coordinates of the direction. When the sketch is planted, conditioning on its noisy norm restricts its direction to a *spherical cap* around the challenge; when it is not planted, the direction is uniform on the whole sphere. Theorem 6.6 bounds how distinguishable the projected cap is after averaging over the random

overlap between the challenge and E. The resulting advantage bound decreases as $1/\sqrt{N}$ and grows only exponentially in k, not in the ambient dimension d.

Why the problem is not easy. The natural attack looks for a sketch whose first k coordinates are close to those of the challenge, and seeing why it fails also explains the shape of the bound. The planted sketch is closer to the challenge than a background sketch does, because the noise moves the revealed coordinates by only a $\beta/4$ fraction of the range those coordinates occupy. But the database gives the attack N chances to find a background sketch that lies that close by accident, and the attack decides correctly only when none does. Each further known plaintext reveals one more coordinate and reduces the rate of these accidental hits by a constant factor, so their expected number falls geometrically in k and grows linearly in N. The attack therefore needs k to grow like $\log N$, which is the rate Theorem 6.6 proves for every distinguisher.

6.9 The Planted Sketch Problem

We define the *planted sketch problem* and bound the advantage of every (potentially unbounded) distinguisher against it unconditionally. Let

$$E = \mathrm{span}(e_1, \ldots, e_k) \subseteq \mathbb{R}^d$$

for the standard basis $e_1, \ldots, e_d$ of $\mathbb{R}^d$, with $E = \{0\}$ when $k = 0$, and let P_E denote orthogonal projection onto E. For $w \in \mathbb{R}^d$, define the randomized sketch

$$\begin{aligned} \mathsf{Sketch}(w): \quad & \eta \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4), \\ & \text{return } \big((w+\eta)[:k], \|w+\eta\|\big) \in \mathbb{R}^k \times \mathbb{R}. \end{aligned}$$

Write $\mu_{d,k,\beta}$ for the distribution of $\mathsf{Sketch}(X)$ when $X \leftarrow\!\!\$\ S^{d-1}$, and write $\nu_{q,d,k,\beta}$ for the distribution of $\mathsf{Sketch}(q)$ for fixed $q \in S^{d-1}$. We omit the subscripts when the parameters are clear. For $N \geq 1$, define the following game:

$\mathbf{Game}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathcal{A})$
$q \leftarrow\!\!\$\ S^{d-1}$
$b \leftarrow\!\!\$\ \{0,1\}$; $i^* \leftarrow\!\!\$\ \{1, \ldots, N\}$
For $i = 1, \ldots, N$:
If $b = 1$ and $i = i^*$: $s_i \leftarrow\!\!\$\ \nu_q$
Else: $s_i \leftarrow\!\!\$\ \mu$
$b' \leftarrow \mathcal{A}\big(q, s_1, \ldots, s_N\big)$
Return $[b = b']$

For a distinguisher $\mathcal{A}$, define

$$\mathbf{Adv}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathcal{A}) = 2\Pr\Big[\mathbf{Game}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathcal{A}) \Rightarrow 1\Big] - 1,$$

with the probability over the challenge bit b, which is uniform, and all other randomness of the game. Let $\delta_{d,k,N,\beta}$ be the statistical distance between the views $(q, s_1, \ldots, s_N)$ under $b = 1$ and $b = 0$. Since b is uniform, the largest value of $\mathbf{Adv}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathcal{A})$ over all, possibly unbounded, distinguishers equals $\delta_{d,k,N,\beta}$. This is the standard identity between advantage against a uniform bit and statistical distance, and it is what makes a bound on $\delta_{d,k,N,\beta}$ a bound on every distinguisher.

6.10 The Chi-Square Divergence

The argument goes through the chi-square divergence. We use it for two reasons. It turns the adversary's uncertainty about which row is planted into an exact factor of $1/N$, and it bounds the statistical distance we want. Let P and Q have densities p and q, and write $P \ll Q$ to mean that P puts no mass where Q puts none. For $P \ll Q$ the density ratio $dP/dQ = p/q$ is defined wherever q is positive, and

$$\chi^2(P \,\|\, Q) = \int \frac{(p-q)^2}{q} = \mathbb{E}_{Y \leftarrow\$ Q}\left[\left(\frac{dP}{dQ}(Y) - 1\right)^2\right],$$

the second expression being the first read as an expectation under Q. It is zero exactly when $P = Q$, and since $\mathbb{E}_Q[dP/dQ] = 1$ it is the variance of the density ratio under Q. It is large when a sample from Q can now and then look far more likely under P, which is what a distinguisher needs.

6.11 The Reduction and the Single-Sketch Bound

Theorem 6.5 (Reduction to the Planted Sketch Problem)**.** *Let $\mathcal{D} = \mathrm{Unif}(S^{d-1})$, let $0 \le k < \min\{n, d\}$, and set $N = n - k$. For every matched adversary $\mathcal{A}$ against $\mathcal{L}_{\mathrm{Gram},\beta}$, there is a planted-sketch distinguisher $\mathsf{R}^{\mathcal{A}}$ such that*

$$\mathbf{Adv}^{\mathsf{MIA}^{\mathsf{match}}}_{\mathcal{L}_{\mathrm{Gram},\beta},\mathcal{D},n,k}(\mathcal{A}) = \mathbf{Adv}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathsf{R}^{\mathcal{A}}).$$

The reduction is efficient whenever $\mathcal{A}$ is.

The proof is in Appendix A.1.

Theorem 6.6 (Single-sketch bound in the useful noise range)**.** *There are absolute constants $c, C > 0$ such that, for all $d > k \ge 1$ and all $\beta \in (0, 2]$,*

$$\mathbb{E}_{q \leftarrow\$ S^{d-1}}\left[\sqrt{\chi^2(\nu_{q,d,k,\beta} \,\|\, \mu_{d,k,\beta})}\right] \le C\left(\frac{c}{\beta}\right)^{k/2}.$$

Consequently, for every $N \ge 1$ and every distinguisher $\mathcal{A}$ in the Planted Sketch Problem,

$$\mathbf{Adv}^{\mathsf{PSP}}_{d,k,N,\beta}(\mathcal{A}) \le \frac{C}{\sqrt{N}}\left(\frac{c}{\beta}\right)^{k/2}.$$

The proof is in Appendix A.3.

7 Secret Projection

We introduce *secret random projection*, following [JL84], for RAP. It applies just as well to the other schemes we consider. Here, the key becomes a single secret map that projects onto a random m-dimensional subspace and rotates within it, and the scheme adds its noise after that as before. For *projected* RAP, fixing $1 \le m \le d$, a key is a uniformly random matrix $Q \in \mathbb{R}^{d \times m}$ with orthonormal columns. On input a unit vector x, the hash algorithm samples fresh noise $\eta \leftarrow\$ \mathcal{B}(0; \beta/4)$ and returns $h(x) = \sqrt{d/m}\, Q^\top x + \eta$. The factor $\sqrt{d/m}$ preserves inner products in expectation before noise. The scheme is given as pseudocode in Figure 5.

Hash function Projected RAP

Parameters: output dimension $1 \le m \le d$ and noise $\beta \ge 0$.

Kg():

1. Sample $Q \in \mathbb{R}^{d \times m}$ uniformly among matrices with orthonormal columns. // uniform secret subspace
2. Return $K = Q$.

$\mathsf{H}_Q(x)$: Input: unit vector $x \in \mathbb{R}^d$.

1. Sample $\eta \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4)$ in $\mathbb{R}^m$. // fresh per evaluation
2. Return $h \leftarrow \sqrt{d/m}\, Q^\top x + \eta$.

Figure 5: Projected RAP. Taking $m = d$ recovers RAP.

7.1 Projected Noisy-Gram Leakage Profile

Definition 7.1 (Projected noisy-Gram leakage profile)**.** Fix $1 \le m \le d$ and $\beta \ge 0$, and let $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$. Sample Q and the noise vectors exactly as the scheme does, set $h_i = \sqrt{d/m}\, Q^\top x_i + \eta_i$, and set

$$\mathcal{L}_{\mathrm{PGram},m,\beta}(S) = \Gamma(h_1, \ldots, h_n).$$

Let $\mathcal{L}^{\mathrm{triv}}_{\mathrm{PRAP},m,\beta}$ denote the trivial leakage profile of projected RAP.

Lemma 7.2 (Projected RAP leakage profile)**.** *The leakage profiles $\mathcal{L}^{\mathrm{triv}}_{\mathrm{PRAP},m,\beta}$ and $\mathcal{L}_{\mathrm{PGram},m,\beta}$ are equivalent.*

The proof follows the proof of Lemma 6.2 with $\mathbb{R}^m$ in place of $\mathbb{R}^d$, and is given in Appendix C.2.

7.2 Projection Moments

Proposition 7.3 (Inner product under secret projection)**.** *Suppose $d \ge 2$, fix $x, y \in \mathbb{R}^d$, and write $c = \langle x, y \rangle$. For $P = QQ^\top$ and $G_{x,y} = (d/m) x^\top P y$,*

$$\mathbb{E}[G_{x,y}] = c$$

and

$$\mathrm{Var}(G_{x,y}) = \frac{(d-m)\big(d\,\|x\|^2\,\|y\|^2 + (d-2)c^2\big)}{m(d-1)(d+2)}.$$

When $d = 1$, necessarily $m = 1$ and the variance is zero. For orthogonal unit vectors the standard deviation is

$$\sqrt{\frac{d(d-m)}{m(d-1)(d+2)}}.$$

The proof is in Appendix C.3.

Discussion. By Lemma 7.2, the adversary sees the Gram matrix of the m-dimensional hash values. By Proposition 7.3, each entry is the true inner product on average, and its variance grows as m shrinks. The entries are therefore noisier than those of the noisy Gram matrix of Section 6.5, which weakens our inner-product-based attacks. The retrieval backend uses the same hash values, so m determines both the attack difficulty and the retrieval loss.

7.3 Attacks on Projected RAP

We specify the two attacks used in the unmatched setting for projected RAP. Each receives the known database plaintexts $x_1, \ldots, x_k$, the challenge p, the shuffled hash values, and the parameters d, m, and β. Both attacks use the beam search of Section 6.6 to retain low-error partial assignments of the known plaintexts to hash values; they differ only in how they score the challenge against a retained assignment.

Joint Gram-pattern attack. This attack adapts the beam-search attack of Section 6.6 to projection. For a retained assignment π and the index j of an unused hash value, it forms the same joint error $T_p(\pi, j)$ between the inner products of the known plaintexts and challenge and those of the assigned hash values. It reports the smallest root-mean-square error over all retained assignments and unused hash values, calling p a member when this score falls below the calibrated threshold. The only change from Section 6.6 is that the inner products among hash values are now estimates of the projected inner products, with the mean of Proposition 7.3.

Conditional residual attack. This attack uses the projection variance, not only the mean. Fix a retained assignment π. Regress the challenge on the known plaintexts by computing

$$a \leftarrow \arg\min_{a'} \left\{ \left\| p - \textstyle\sum_i a'_i x_i \right\|^2 + \lambda \left\| a' \right\|^2 \right\}, \qquad \lambda = 10^{-2}.$$

For a candidate challenge hash value y_j, the same coefficients predict $\langle y_j, y_{\pi(\ell)} \rangle$ as $\sum_i a_i \langle y_{\pi(i)}, y_{\pi(\ell)} \rangle$ for each known index ℓ, and $\|y_j\|^2$ as $\sum_i a_i \langle y_{\pi(i)}, y_j \rangle$. The attack compares each prediction with the value expected for a true challenge hash value and divides by the predicted standard deviation, so a hash value matching within the predicted spread gives a small score. Projection moments come from Proposition 7.3. The noise calculation uses coordinate variance $\beta^2/(16(m+2))$, approximates the uniform ball noise by Gaussian moments, and treats the residual components separately rather than modeling the correlations from their shared noise. The attack reports the smallest standardized score over retained assignments and unused hash values and thresholds it, so this is a heuristic score rather than an analytic likelihood ratio.

Neither attack recovers the secret projection or the hidden matching, and each tests the challenge jointly against the retained assignments.

7.4 Attack and Utility Results

We evaluate the unmatched attack of Section 6.6 against projected RAP, in two forms given in Section 7.3: one scores assignments with the projected inner products as in Section 6.6, and one also uses the projection variance of Proposition 7.3. Both compare the challenge against the stored items through their relations to the known plaintexts. At $m = 256$ the unmatched attack reaches a breakpoint at 18 known plaintexts rather than 5 at $n = 100$, and at 128 rather than 12 at $n = 1{,}000$, so, under our fixed beam-search budget, the number of known plaintexts projection adds to the breakpoint grows with the database. Retrieval quality falls as well. At $m = 256$ on a database

of one million vectors the scheme retains 98.3% of the plaintext mean reciprocal rank at $\beta = 0.5$. The attack sweeps and the retrieval measurements at each database size are in Appendix C.5, and Appendix C.4 has a *task-aware* variant that recovers some of this utility.

We compare output dimensions at $\beta = 0.5$ in Table 1. The attack experiment uses $n = 100$, while Neighbor Recall@10 is measured at $n = 10{,}000$. The projection SD is the standard deviation in Proposition 7.3 for orthogonal unit vectors and excludes the added noise. The symbol > 99 means only that the attack did not reach advantage 0.5 through the largest tested k.

Output dimension m	Projection SD (c=0)	Neighbor Recall@10	First k with advantage ≥ 0.5
384 (unprotected)	—	1.000	—
384	0.000	0.929	5
256	0.036	0.767	18
192	0.051	0.689	36
128	0.072	0.586	> 99

Table 1: Projected RAP on MS-MARCO at $\beta = 0.5$, with the attack breakpoint measured at $n = 100$. The largest standard error at a reported breakpoint is 0.059.

8 Comparison-Only Leakage and Ideal DCRE

We move beyond DCPE/DCPH to the comparison-only leakage profile underlying ideal *distance-comparison revealing encryption* (DCRE) as introduced by FGHO [FGHO22]. This profile discards coordinates and numerical distances and keeps only distance orderings. A DCRE scheme admits an arbitrary efficient comparison algorithm rather than literal distance comparison on ciphertexts, and thus is not subject to FGHO's result that any DCPE scheme must approximately preserve distances themselves, not only their comparisons [FGHO22]. This is our only *hypothetical* leakage profile, since FGHO give no construction. Our goal is therefore to determine whether such a scheme is worth pursuing.

8.1 Leakage Profiles

Definition 8.1 (Comparison-only leakage profile)**.** Fix $\beta \geq 0$ and let $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$. Sample $\eta'_1, \ldots, \eta'_n \leftarrow\!\!\$\ \mathcal{B}(0; \beta/4)$ independently and set $y_i = x_i + \eta'_i$. For each i, let r_i be the ordering of $\{1, \ldots, n\} \setminus \{i\}$ that lists the indices j in increasing order of $\mathsf{dist}(y_i, y_j)$, breaking ties independently and uniformly. We define

$$\mathcal{L}_{\mathrm{cmp},\beta}(S) = (r_1, \ldots, r_n).$$

When β is fixed by the experiment, we write $\mathcal{L}_{\mathrm{cmp}}$ for short. Under a relabeling π, row r_i becomes row $r_{\pi(i)}$, and every position j appearing in that row is replaced by $\pi(j)$.

Lemma 8.2 (Comparison leakage from Gram leakage)**.** *$\mathcal{L}_{\mathrm{cmp},\beta}$ is simulatable from $\mathcal{L}_{\mathrm{Gram},\beta}$.*

Proof. For every pair i, j, compute the noisy squared distance from the Gram matrix as

$$\mathsf{dist}(y_i, y_j)^2 = G_{ii} + G_{jj} - 2G_{ij}.$$

Sort these distances separately in each row and break ties randomly as specified in Definition 8.1. This is exactly the distribution of $\mathcal{L}_{\mathrm{cmp},\beta}(S)$. □

By Lemma 6.2, RAP's leakage profile is equivalent to noisy Gram leakage, so by Lemma 8.2 comparison-only leakage is simulatable from RAP's. An attack on comparison-only leakage is therefore an attack on RAP, and a security bound for RAP holds for comparison-only leakage. In particular, for $\mathcal{D} = \mathrm{Unif}(S^{d-1})$ and $0 < \beta \leq 2$, Theorem 6.4 bounds the advantage of every matched, and so every unmatched, adversary against $\mathcal{L}_{\mathrm{cmp},\beta}$ by $(C/\sqrt{n-k})(c/\beta)^{k/2}$. The implication is one-way. We do not prove that comparison-only leakage is strictly less informative, and the inherited bound need not be tight; the matched breakpoints we measure for the two profiles at $n = 100$ differ by one known plaintext. The RAP attacks themselves do not apply here, since they use the coordinates and numerical distances this profile discards.

At $\beta = 0$ the comparison-only leakage profile is ideal DCRE in the sense of FGHO [FGHO22], since evaluating the comparison predicate on all triples determines $\mathcal{L}_{\mathrm{cmp},0}$ up to tie-breaking. We restrict to unit vectors so that distance and inner-product orderings agree and all our leakage profiles use the same geometry. The noiseless experiment therefore also covers ideal inner-product-comparison encryption, which Acharya et al. [AFLO25] pose as an open problem; the security theorem for their access-controlled construction lets the simulator see exactly these orderings, so that theorem on its own does not give membership privacy. For $\beta > 0$ the profile is not ideal DCRE, but it lets us compare with RAP at the same noise level.

The projected variant. For a controlled comparison with Section 7, define $\mathcal{L}_{\mathrm{PCmp},m,\beta}$. Sample $Q \in \mathbb{R}^{d\times m}$ uniformly with orthonormal columns and independent $\eta_i \leftarrow\!\!\$\ \mathcal{B}(0;\beta/4)$, set

$$z_i = \sqrt{d/m}\, Q^\top x_i + \eta_i,$$

and return the row-wise distance orderings of $(z_1, \ldots, z_n)$ with the tie-breaking rule of Definition 8.1. Position i labels z_i and its row, and a relabeling acts on both the rows and the positions appearing within them.

Lemma 8.3 (Projected comparison leakage)**.** *$\mathcal{L}_{\mathrm{PCmp},m,\beta}$ is simulatable from $\mathcal{L}_{\mathrm{PGram},m,\beta}$.*

Proof. Apply the same distance calculation and row-wise sorting as in the proof of Lemma 8.2 to the projected Gram matrix. □

At $\beta = 0$, the projected comparison-only leakage profile is ideal DCRE on the projected representations rather than on the original vectors. The distinction matters because projection can reverse distance comparisons, so these orderings need not be those of the original vectors. At fixed (m, β), projected RAP and this leakage profile use the same projected, perturbed representation and therefore have the same retrieval quality.

8.2 Rank-Based Attacks on Comparison-Only Leakage

Attack from ranks. The attack follows the unmatched RAP attack's search over assignments of known vectors to positions, but it replaces numerical inner-product agreement with rank agreement. The known database vectors predict their pairwise positions in the orderings, while the challenge predicts how a member position should rank relative to those same vectors.

Rank residuals. Write R_{ab} for the normalized rank that position b holds in position a's ordering in $\mathcal{L}_{\mathrm{cmp}}$, so that $R_{ab} = (\mathrm{rank}_a(b) + 1/2)/(n-1)$, where $\mathrm{rank}_a(b) \in \{0, \ldots, n-2\}$ is the position of b in a's ordering. It is close to 0 when b is a's nearest position and close to 1 when b is the farthest.

From a public reference set P the adversary estimates the rank one vector should hold in another's ordering as

$$\widehat{R}(x,y) = \frac{|\{z \in P : \mathsf{dist}(x,z) < \mathsf{dist}(x,y)\}| + 1/2}{|P|+1}.$$

When $\beta > 0$ it averages the numerator over the public noise simulation. Write $\widehat{R}_{ij}$ for the predicted rank of x_j in x_i's ordering, and $\widehat{R}_{ip}$, $\widehat{R}_{pi}$ for the two ranks between x_i and the challenge p. A partial assignment π maps known vectors to distinct positions, and its error over the assigned pairs is $E(\pi) = \sum_{i \neq j} (R_{\pi(i)\pi(j)} - \widehat{R}_{ij})^2$. For a complete assignment and an unassigned position c, the challenge residual is

$$D(\pi, c) = \sum_{i=1}^{k} \Big[\big(R_{\pi(i)\,c} - \widehat{R}_{ip}\big)^2 + \big(R_{c\,\pi(i)} - \widehat{R}_{pi}\big)^2 \Big].$$

Matched attack. In the matched setting, the adversary is additionally supplied the position $\pi(i)$ corresponding to each known plaintext x_i; this correspondence is auxiliary known-plaintext information, not part of the comparison-only leakage profile. The attack skips the beam search and uses

$$\min_{c \notin \pi(\{1,\ldots,k\})} \sqrt{\frac{D(\pi,c)}{2k}}$$

as its challenge score, with the threshold selected on separate calibration games as below.

Beam search. In the unmatched game the attack runs the beam search of Section 6.6, keeping the B partial assignments of smallest error E. The known vectors are ordered so that the most unusual rank constraints enter before the beam is pruned. The attack then tests a prefix of the final beam, selecting the prefix width and the threshold on calibration runs and holding both fixed thereafter. Write $\Pi_k^{(w)}$ for the selected prefix. The membership score is the smallest challenge residual over it,

$$\widehat{s}(p) = \sqrt{\frac{1}{2k} \min_{\substack{\pi \in \Pi_k^{(w)} \\ c \notin \pi(\{1,\ldots,k\})}} D(\pi, c)},$$

so the known-vector error E only ranks the assignments, and it is not added into the challenge score.

Running time. The ordering heuristic compares each of $O(k^2)$ known-vector pairs with $n(n-1)$ ordered position pairs. After that, the first beam step examines $n(n-1)$ placements, while each later step examines at most Bn extensions and adds $O(k)$ rank constraints per extension. Including the final challenge test, the attack uses $O(k^2n^2 + Bnk^2)$ arithmetic operations.

Retrieval metric. We use the top-10 Neighbor Recall metric of Section 2 on the normalized MS-MARCO embeddings; unlike MRR@10 and the metrics of Appendix C.4, it uses no relevance labels.

8.3 Projected Comparison-Only Results

After projection, the matched attack still reaches a breakpoint at every tested dimension, while the unmatched attack does not reach one at any $m < 384$. Table 3 contains the breakpoints, where $> k$

means only that the attack did not reach advantage 0.5 through the largest tested k, together with the fraction of distance comparisons the projected representation preserves, measured at $\beta = 0$ to isolate the cost of projection. We compare the two profiles at fixed $k = 32$ in Table 2. Since the two profiles share a projection and a perturbation at each m, they have the same retrieval quality, so their breakpoints compare directly. At $m = 256$, where Neighbor Recall@10 is 0.767 for both, the attack on projected RAP reaches a breakpoint at $k = 18$ while our attack on the comparison-only profile reaches no breakpoint through $k = 99$. Withholding coordinates and numerical distances therefore multiplies what the tested unmatched attack needs by at least five, at no cost in retrieval quality.

Noise	Output dimension	Neighbor	Projected RAP		Comparison-only	
β	m	Recall@10	Advantage	Nonmember	Advantage	Nonmember
0	384	1.000	0.997 ± 0.003	0.003	1.000 ± 0.000	0.000
	256	0.779	0.963 ± 0.011	0.017	0.151 ± 0.030	0.409
	192	0.701	0.540 ± 0.020	0.071	0.011 ± 0.041	0.417
	128	0.592	0.006 ± 0.027	0.594	0.023 ± 0.023	0.500
0.5	384	0.929	0.994 ± 0.006	0.006	0.986 ± 0.005	0.000
	256	0.767	0.937 ± 0.016	0.009	0.114 ± 0.031	0.371
	192	0.689	0.417 ± 0.035	0.089	0.014 ± 0.023	0.431
	128	0.586	0.003 ± 0.037	0.520	0.006 ± 0.045	0.320

Table 2: Controlled unmatched attacks on MS-MARCO at $k = 32$, with advantage and the nonmember acceptance rate measured as described in Appendix F.1. The original dimension is 384, and m is the output dimension after projection. One Neighbor Recall@10 column serves both leakage profiles. The comparison-only column at $\beta = 0$ is projected ideal DCRE.

		$\beta = 0$ (ideal DCRE)		$\beta = 0.5$	
m	Comparisons preserved	Matched	Unmatched	Matched	Unmatched
384	1.00	3	5	4	8
256	0.82	12	> 99	12	> 99
192	0.76	16	> 99	16	> 99
128	0.71	32	> 99	32	> 99

Table 3: Breakpoints for matched and unmatched attacks on the comparison-only leakage profile of a projected representation in m dimensions, MS-MARCO with $n = 100$.

Three rows of Table 3 reach no breakpoint. At $\beta = 0.5$ and $k = 99$, their advantages are 0.411 ± 0.035 at $m = 256$ and still rising, 0.111 ± 0.026 at $m = 192$, and 0.014 ± 0.024 at $m = 128$, which is indistinguishable from zero. These three rows differ in kind. At $m = 256$ the nonmember acceptance rate is 0.04 and the AUC is 0.644, so the attack accepts few nonmembers and is limited by how few members it accepts. At $m = 128$ the nonmember acceptance rate is 0.55 and the AUC is 0.485, so the attack accepts about half of everything it is shown and its score has no ordering information. At $m = 192$ the AUC is 0.539, barely above chance. At $n = 1{,}000$ the same sweep reaches no breakpoint at any of the three output dimensions, at either noise level. At $k = 99$ the advantages are at most 0.034 and every AUC lies between 0.49 and 0.52, so the scores order members no better than chance. The additional experiments at full dimension are in Appendix D,

and the run counts, seeds, and beam widths are in Appendix F.1.

9 Encrypted HNSW Index Leakage and Attacks

We now consider the snapshot leakage of an owner-built encrypted HNSW index built using function-hiding inner-product functional encryption (FH-IPFE) [KLM+18], which exposes less of the plaintext geometry than the prior approaches. FH-IPFE encrypts each database vector x_i as a ciphertext ct_{x_i} and represents a query q by a functional key sk_q, and evaluation on $(\mathsf{ct}_{x_i}, \mathsf{sk}_q)$ returns $\langle x_i, q\rangle$. There is no evaluation algorithm for two database ciphertexts, so a server holding only ciphertexts learns nothing until a query key is issued. The simplest use is a linear scan: the server evaluates sk_q against every ciphertext and returns the largest inner products. This has no snapshot leakage beyond the ciphertexts, but is prohibitive at production scale.

To obtain sublinear search, Acharya, Feng, Langrehr, and O'Neill (AFLO) [AFLO25] observe that the index must be built by the data owner, since building it needs distances between database vectors. The owner sends the index state to the server with the encrypted database, and the server traverses it at query time using the query key. The index state depends on the database, so the snapshot now leaks information. We study HNSW indexes [MY20]. AFLO also consider HNSW in this setting and specify the leakage of their protocol, but do not study whether the leaked index state supports membership inference.

9.1 HNSW Index Leakage Profile

To make the leakage profile independent of implementation details, we formalize an idealized HNSW constructor, which takes the squared-distance matrix rather than the vectors, computes in exact arithmetic, and reads all of its randomness from a public tape. For a database $S = (x_1, \ldots, x_n)$, let

$$\Delta(S)_{ij} = \|x_i - x_j\|_2^2$$

be its squared-distance matrix. Fix a public algorithm $\mathsf{HNSWBuild}_{\theta_{\mathrm{H}}}$ that, on a squared-distance matrix and a random tape ω, runs the HNSW construction in exact arithmetic and returns its final layered directed graph. The public parameters θ_{H} fix all construction parameters, including `M` and `efConstruction`; the tape ω is drawn from a distribution $\mathcal{R}_{\theta_{\mathrm{H}},n}$ that depends only on θ_{H} and n, and every random choice the constructor makes, such as level assignments and tie breaking, is read from it. Vertices are numbered $1, \ldots, n$ in the row order of the input matrix, which is the insertion order. Our experiments use Faiss HNSW on the vectors themselves; the leakage profile and reduction below concern this idealized constructor, and we do not claim equivalence to Faiss's serialization, memory layout, or floating-point behavior.

As with every leakage profile in this paper, vertices are labeled by positions in the protected database, which identify stored items but not the plaintexts that produced them. The correspondence between known plaintexts and vertices is given to the adversary only in the matched game.

The stored snapshot also contains the FH-IPFE ciphertexts, which we omit from the profile. This rests on one assumption: the snapshot is taken before any functional key is issued, so against a computationally bounded adversary the ciphertexts are simulatable from the public parameters and n by function-hiding security. Under that assumption, the distributional identities we prove for the index state become computational indistinguishability for the full snapshot.

Definition 9.1 (HNSW index leakage profile)**.** Let $S = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$ and set $\Delta = \Delta(S)$. Sample a hidden insertion permutation $\tau \leftarrow\!\!\$\ \mathsf{Perm}(n)$ and construction randomness $\omega \leftarrow\!\!\$\ \mathcal{R}_{\theta_{\mathrm{H}},n}$,

independently of each other and of S. Define the permuted distance matrix by

$$\Delta^{\tau}_{ab} = \Delta_{\tau^{-1}(a),\tau^{-1}(b)} \qquad (a, b \in [n]),$$

and run

$$S_\tau \leftarrow \mathsf{HNSWBuild}_{\theta_{\mathrm{H}}}(\Delta^{\tau}; \omega).$$

Here vertex a in S_τ represents the record inserted in position a, namely $x_{\tau^{-1}(a)}$. Relabel vertex $\tau(i)$ by database position i. Let $\mathrm{lev}_\tau(i)$ be the maximum level of the resulting vertex i, let $L_\tau = \max_i \mathrm{lev}_\tau(i)$, and for each $r \in \{0, \ldots, L_\tau\}$ define

$$V_{\tau,r} = \{i \in [n] : \mathrm{lev}_\tau(i) \geq r\}.$$

Let $E_{\tau,r} \subseteq \{(i,j) \in V_{\tau,r} \times V_{\tau,r} : i \neq j\}$ be the directed neighbor relation stored at level r, and let $e_\tau \in V_{\tau,L_\tau}$ be the entry-point vertex. Return

$$\mathcal{L}_{\mathrm{HNSW},\theta_{\mathrm{H}}}(S) = \big(\mathrm{lev}_\tau, (E_{\tau,r})_{r=0}^{L_\tau}, e_\tau\big).$$

Neither τ nor ω is returned.

Thus, the leakage profile exposes the final layered directed graph, including which vertices occur at which levels and the entry point, but not the history that produced it. Our concrete attacks below use only the edge relations and level structure, so failure of one of them is not a security result for the full profile.

For this idealized HNSW constructor, the state is efficiently simulatable from the noiseless Gram matrix. Indeed, given G, the simulator computes $\Delta_{ij} = G_{ii} + G_{jj} - 2G_{ij}$ and runs the randomized HNSW constructor. Thus the HNSW state is no more informative than the Gram matrix, but this implication is one-way: the graph need not reveal the Gram matrix. So building an HNSW index over RAP hash values reveals nothing beyond what the hash values already reveal. Lemma E.1 states this precisely, and the proof appears in Appendix E.1.

9.2 Matched and Unmatched Attacks on the Index

We give two attacks, one matched and one unmatched. Both use the fact that the final graph retains some information about the geometry from which it was constructed; namely, nearby vectors are more likely to appear in one another's neighbor lists than unrelated vectors. The matched attacker is additionally told which vertices hold its known plaintexts. The unmatched attacker receives the same HNSW snapshot but not that correspondence, and must infer a useful placement of the known plaintexts.

The edge model. The adversary turns this geometric dependence into a score by fitting two quantities on public databases. The first is $p_{\mathrm{edge}}(t)$, the probability that the directed neighbor list of one vertex contains another when their vectors have inner product t. The second is the baseline rate p_0 of a directed edge between an unrelated pair. We fit these separately at each layer and add the layer scores, so what follows is the score for one layer. Write $e_{a\to b} = 1$ when the neighbor list of a contains b, and 0 otherwise. One observed bit at predicted inner product t contributes

$$\lambda_t(e) = e \log \frac{p_{\mathrm{edge}}(t)}{p_0} + (1 - e) \log \frac{1 - p_{\mathrm{edge}}(t)}{1 - p_0}.$$

This contribution is positive when the observed edge or nonedge is more likely at inner product t than for an unrelated ordered pair. Since each vertex keeps only a bounded neighbor list, the edge events are dependent, so this is a fitted score rather than an exact likelihood ratio.

Matched attack. In the matched known-plaintext model, the adversary is additionally given the vertex v_i corresponding to each known plaintext x_i. This correspondence is not part of $\mathcal{L}_{\mathrm{HNSW},\theta_{\mathrm{H}}}$. The adversary knows the challenge vector p, so it can compute $t_i = \langle p, x_i \rangle$, the inner product that a vertex holding p would have with each known plaintext. For any other vertex u, it reads the $2k$ directed bits between u and the known vertices and asks how well they match what each t_i predicts. Let U be the vertices other than $v_1, \ldots, v_k$. The evidence that u holds the challenge is

$$\ell(u) = \sum_{i=1}^{k} \big[\lambda_{t_i}(e_{u \to v_i}) + \lambda_{t_i}(e_{v_i \to u})\big].$$

The adversary does not know which vertex holds the challenge. Rather than select the best-fitting vertex after observing the graph, it weights every candidate equally and reports

$$\mathrm{score}(p) = \log\Big(\tfrac{1}{|U|} \sum_{u \in U} e^{\ell(u)}\Big).$$

Unmatched attack. In the unmatched model, the adversary knows the plaintext vectors $x_1, \ldots, x_k$ but not their graph vertices. It therefore has to place the known plaintexts before it can apply the matched score. We use the beam search from our other unmatched attacks, specialized to graph edges. A partial placement π maps a subset I of the known plaintexts to distinct vertices. We score its plausibility from the inner products among the known plaintexts:

$$L(\pi) = \sum_{\substack{i,j \in I \\ i<j}} \big[\lambda_{\langle x_i, x_j \rangle}(e_{\pi(i) \to \pi(j)}) + \lambda_{\langle x_i, x_j \rangle}(e_{\pi(j) \to \pi(i)})\big].$$

The search keeps the B partial placements with largest L at each depth. For every complete placement π in the final beam $\mathcal{P}_B$, it computes the matched score under that placement, denoted $s_\pi(p)$, and reports

$$s_{\mathrm{best}}(p) = \max_{\pi \in \mathcal{P}_B} \{L(\pi) + s_\pi(p)\} - \max_{\pi \in \mathcal{P}_B} L(\pi).$$

This measures how much the best joint fit improves when the challenge is added. The attack need not output, or even recover, the true placement. The known-plaintext ordering rule, running times, and controls are in Appendix E.

9.3 Experimental Results

We build Faiss HNSW indexes with `M` = 32 and `efConstruction` = 100 on MS-MARCO, SciFact, and data uniform on the sphere, fit p_{edge} and p_0 on separate public databases, and run five seeds per setting. Our matched edge-based attack does extract membership information from the graph, though its strength varies with the data distribution, and we do not have an explanation for the gap between SciFact and MS-MARCO. We fix the score direction and threshold in advance at every setting, on calibration games separate from the held-out games we score, so the numbers below are MIA advantage as everywhere else in this paper.

On MS-MARCO the breakpoint is $k = 128$, 192, 384, 1,024 and 3,200 at $n = 500$, 1,000, 2,000, 5,000 and 10,000, which is between 19% and 32% of the database with no downward trend. At those breakpoints the attack accepts between 0.750 and 0.816 of members and between 0.142 and 0.254 of nonmembers, with AUC between 0.839 and 0.914. On data uniform on the sphere the attack is stronger at every size, with breakpoints $k = 64$, 96, 128, 224 and 512, falling from 12.8%

n	k	Advantage	M/N accept	AUC	Prev. k
MS-MARCO					
500	128	0.648 ± 0.013	0.790/0.142	0.914	96 (0.486)
1,000	192	0.582 ± 0.012	0.816/0.234	0.888	160 (0.466)
2,000	384	0.544 ± 0.039	0.750/0.206	0.860	320 (0.472)
5,000	1,024	0.510 ± 0.015	0.764/0.254	0.839	896 (0.474)
10,000	3,200	0.540 ± 0.027	0.764/0.224	0.858	2,560 (0.414)
Uniform sphere					
500	64	0.614 ± 0.022	0.786/0.172	0.915	48 (0.462)
1,000	96	0.652 ± 0.026	0.802/0.150	0.921	64 (0.394)
2,000	128	0.604 ± 0.030	0.758/0.154	0.896	96 (0.414)
5,000	224	0.508 ± 0.035	0.670/0.162	0.852	192 (0.388)
10,000	512	0.602 ± 0.034	0.770/0.168	0.894	384 (0.452)

Table 4: Matched attack on the encrypted HNSW index, with the score direction and threshold fixed in advance at every setting on separate calibration games. Each row is the first tested k at which mean advantage reaches 0.5. "M/N accept" gives member and nonmember acceptance. The last column gives the preceding tested k and its advantage. Five seeds per setting; errors are standard errors across seeds. On SciFact at $n = 2{,}000$ the attack does not reach a breakpoint through $k = 512$, where its advantage is 0.100 ± 0.027.

to 5.1% of the database. On SciFact at $n = 2{,}000$ it does not reach a breakpoint through $k = 512$, where its advantage is 0.100. Every setting is in Table 4.

In the unmatched setting the attack fails outright, recovering known vertices within one percent of the random-set baseline even with hundreds of known plaintexts. This is the widest matched/unmatched gap we observe, and it is a negative result about our attack rather than a security result for HNSW. The setup, the full sweeps, the null reference obtained by swapping member and nonmember labels, and the unmatched measurements are in Appendix E.4.

References

[AAG25] Maya Anderson, Guy Amit, and Abigail Goldsteen. Is my data in your retrieval database? Membership inference attacks against retrieval augmented generation. In *International Conference on Information Systems Security and Privacy (ICISSP)*, pages 474–485, 2025. https://doi.org/10.5220/0013108300003899.

[AFLO25] Ojaswi Acharya, Weiqi Feng, Roman Langrehr, and Adam O'Neill. Access-controlled inner product function-revealing encryption. *IACR Communications in Cryptology (CIC)*, 2(3), 2025. https://cic.iacr.org/p/2/3/7.

[BBG+11] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. Noiseless database privacy. In *ASIACRYPT*, pages 215–232, 2011.

[BBO07] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552, 2007.

[BCLO09] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 224–241, 2009.

[BCO11] Alexandra Boldyreva, Nathan Chenette, and Adam O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *LNCS*, pages 578–595, 2011.

[BGKS13] Raef Bassily, Adam Groce, Jonathan Katz, and Adam Smith. Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

[BLR+15] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In *Advances in Cryptology – EUROCRYPT 2015*, volume 9057 of *LNCS*, pages 563–594, 2015.

[BLV19] Elette Boyle, Rio LaVigne, and Vinod Vaikuntanathan. Adversarially robust property-preserving hash functions. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:20. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

[BT21] Alexandra Boldyreva and Tianxin Tang. Privacy-preserving approximate k-nearest-neighbors search that hides access, query and volume patterns. *Proceedings on Privacy Enhancing Technologies*, 2021(4):549–574, 2021. Full version: `https://eprint.iacr.org/2021/816`.

[Bue26] Alex Soto Bueno. Protecting AI embedding vectors by using approximate distance preserving encryption for RAG applications. `https://developer.ibm.com/articles/java-vector-embeddings-encryption/`, 2026.

[CCD+20] Hao Chen, Ilaria Chillotti, Yihe Dong, Oxana Poburinnaya, Ilya Razenshteyn, and M. Sadegh Riazi. SANNS: Scaling up secure approximate k-nearest neighbors search. In *USENIX Security Symposium*, pages 2111–2128, 2020.

[CCN+22] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy (S&P)*, 2022.

[CGPR15] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In *ACM Conference on Computer and Communications Security (CCS)*, pages 668–679, 2015.

[CK16] Daniel Cullina and Negar Kiyavash. Improved achievability and converse bounds for Erdős–Rényi graph matching. In *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, pages 63–72, 2016.

[CLDW26] Haoyu Cui, Zengpeng Li, Tien Tuan Anh Dinh, and Mei Wang. MESS: Fast and private semantic search on multi-graph HNSW, 2026. arXiv:2607.28999.

[CZW+21] Qi Chen, Bing Zhao, Haidong Wang, Mingqin Li, Chuanjie Liu, Zengzhong Li, Mao Yang, and Jingdong Wang. SPANN: Highly-efficient billion-scale approximate nearest neighbor search. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 5199–5212, 2021.

[DCK19] Osman Emre Dai, Daniel Cullina, and Negar Kiyavash. Database alignment with Gaussian features. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3225–3233, 2019.

[DDC16] F. Betül Durak, Thomas M. DuBuisson, and David Cash. What else is revealed by order-revealing encryption? In *ACM Conference on Computer and Communications Security (CCS)*, pages 1155–1166, 2016.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, pages 265–284, 2006.

[DRS22] Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(1):3–37, 2022.

[FGHO22] Georg Fuchsbauer, Riddhi Ghosal, Nathan Hauke, and Adam O'Neill. Approximate distance-comparison-preserving symmetric encryption. In *Security and Cryptography for Networks (SCN)*, volume 13409 of *Lecture Notes in Computer Science*, pages 117–144, 2022. Full version: `https://eprint.iacr.org/2021/1666`.

[FLS21] Nils Fleischhacker, Kasper Green Larsen, and Mark Simkin. Property-preserving hash functions for hamming distance from standard assumptions. Cryptology ePrint Archive, Paper 2021/793, 2021. `https://eprint.iacr.org/2021/793`. Also in EUROCRYPT 2022.

[FMWX20] Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. Spectral graph matching and regularized quadratic relaxations: Algorithm and theory. In *International Conference on Machine Learning (ICML)*, pages 2985–2995, 2020.

[FS21] Nils Fleischhacker and Mark Simkin. Robust property-preserving hash functions for hamming distance and more. In *Advances in Cryptology — EUROCRYPT 2021*, volume 12698 of *Lecture Notes in Computer Science*, pages 311–337, 2021.

[GL24] Jianyang Gao and Cheng Long. RaBitQ: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search. *Proceedings of the ACM on Management of Data*, 2(3), 2024.

[GLK13] Chris Giannella, Kun Liu, and Hillol Kargupta. Breaching Euclidean distance-preserving data perturbation using few known inputs. *Data & Knowledge Engineering*, 83:93–110, 2013. `https://doi.org/10.1016/j.datak.2012.10.004`.

[GSB+16] Paul Grubbs, Kevin Sekniqi, Vincent Bindschaedler, Muhammad Naveed, and Thomas Ristenpart. Leakage-abuse attacks against order-revealing encryption. Cryptology ePrint Archive, Paper 2016/895, 2016. `https://eprint.iacr.org/2016/895`.

[GSL+20] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *International Conference on Machine Learning (ICML)*, volume 119, pages 3887–3896, 2020.

[HLTW22] Justin Holmgren, Minghao Liu, LaKyah Tyner, and Daniel Wichs. Nearly optimal property preserving hashing. Cryptology ePrint Archive, Paper 2022/842, 2022. `https://eprint.iacr.org/2022/842`. Also in CRYPTO 2022.

[HRBLM07] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts Amherst, 2007.

[HSR+08] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V. Pearson, Dietrich A. Stephan, Stanley F. Nelson, and David W. Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genetics*, 4(8):e1000167, 2008.

[HSS+22] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys*, 54(11s):1–37, 2022.

[HTH+24] Yu-Hsiang Huang, Yuche Tsai, Hsiang Hsiao, Hong-Yi Lin, and Shou-De Lin. Transferable embedding inversion attack: Uncovering privacy risks in text embeddings without model queries. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4193–4205, 2024.

[IKK12] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *Network and Distributed System Security Symposium (NDSS)*, 2012.

[Iro24] IronCore Labs. Cloaked AI: Encrypt data in vector databases. `https://ironcorelabs.com/products/cloaked-ai/`, 2024.

[JDJ21] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021.

[JKMP26] Charanjit S. Jutla, Dhilung Kirat, Nathan Manohar, and Sikhar Patranabis. Distance-preserving encryption for natural-language embedding vectors. Unpublished manuscript, cited with the authors' permission, 2026.

[JL84] William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

[JP18] Marc Joye and Alain Passelègue. Function-revealing encryption: Definitions and constructions. In *Security and Cryptography for Networks*, volume 11035 of *LNCS*, pages 527–543. Springer, 2018.

[JWK+21] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021(2):348–368, 2021.

[JZSM25] Rishi Jha, Collin Zhang, Vitaly Shmatikov, and John X. Morris. Harnessing the universal geometry of embeddings. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 45963–45987. Curran Associates, Inc., 2025.

[KIK12] Mehmet Kuzu, Mohammad Saiful Islam, and Murat Kantarcioglu. Efficient similarity search over encrypted data. In *IEEE International Conference on Data Engineering (ICDE)*, pages 1156–1167, 2012.

[KKNO16] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1329–1340, 2016.

[KLM+18] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J. Wu. Function-hiding inner product encryption is practical. In *Security and Cryptography for Networks*, volume 11035 of *LNCS*, pages 544–562. Springer, 2018. Full version: `https://eprint.iacr.org/2016/440`.

[KOM+20] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

[KOV15] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International Conference on Machine Learning (ICML)*, pages 1376–1385, 2015.

[KPRdV26] Anne-Marie Kermarrec, Rafael Pires, Mathis Randl, and Martijn de Vos. A unified benchmark for privacy-preserving vector search. arXiv preprint arXiv:2608.01192, 2026. `https://arxiv.org/abs/2608.01192`.

[KPT19] Evgenios M. Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. Data recovery on encrypted databases with k-nearest neighbor query leakage. In *IEEE Symposium on Security and Privacy (SP)*, pages 1033–1050, 2019. Also as Cryptology ePrint Archive, Paper 2018/719.

[LFP14] Vince Lyzinski, Donniell E. Fishkind, and Carey E. Priebe. Seeded graph matching for correlated Erdős–Rényi graphs. *Journal of Machine Learning Research*, 15(108):3693–3720, 2014.

[LLWY25] Yuying Li, Gaoyang Liu, Chen Wang, and Yang Yang. Generating is believing: Membership inference attacks against retrieval-augmented generation. In *2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755, 2014.

[Low76] Bruce T. Lowerre. *The HARPY Speech Recognition System*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1976.

[LQS+13] Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. Membership privacy: A unifying framework for privacy definitions. In *ACM Conference on Computer and Communications Security (CCS)*, pages 889–900, 2013.

[LW16] Kevin Lewi and David J. Wu. Order-revealing encryption: New constructions, applications, and lower bounds. In *ACM Conference on Computer and Communications Security*, pages 1167–1178, 2016.

[LXS23] Haoran Li, Mingshi Xu, and Yangqiu Song. Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 14022–14040. Association for Computational Linguistics, 2023.

[LZL25] Mingrui Liu, Sixiao Zhang, and Cheng Long. Mask-based membership inference attacks for retrieval-augmented generation. In *Proceedings of the ACM on Web Conference (WWW)*, pages 2894–2907, 2025.

[LZX+25] Yingfan Liu, Yandi Zhang, Jiadong Xie, Hui Li, Jeffrey Xu Yu, and Jiangtao Cui. Privacy-preserving approximate nearest neighbor search on high-dimensional data. In *41st IEEE International Conference on Data Engineering (ICDE)*, pages 3017–3029, 2025.

[Mic] Microsoft. Data, privacy, and built-in protections in azure AI search. `https://learn.microsoft.com/en-us/azure/search/search-security-built-in`. Accessed August 25, 2026.

[MKSR23] John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12448–12460. Association for Computational Linguistics, 2023.

[MY20] Yu A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020.

[NKW15] Muhammad Naveed, Seny Kamara, and Charles V. Wright. Inference attacks on property-preserving encrypted databases. In *ACM CCS*, 2015.

[NPS+25] Ali Naseh, Yuefeng Peng, Anshuman Suri, Harsh Chaudhari, Alina Oprea, and Amir Houmansadr. Riddle me this! stealthy membership inference for retrieval-augmented generation. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1245–1259, 2025.

[NRS+16] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In *NeurIPS Workshop on Cognitive Computation*, 2016.

[ONN+24] Hiroyuki Ootomo, Akira Naruse, Corey Nolet, Ray Wang, Tamas Feher, and Yong Wang. CAGRA: Highly parallel graph construction and approximate nearest neighbor search for GPUs. In *IEEE International Conference on Data Engineering (ICDE)*, pages 4236–4247, 2024.

[PG11] Pedram Pedarsani and Matthias Grossglauser. On the privacy of anonymized networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1235–1243, 2011.

[PR12] Omkant Pandey and Yannis Rouselakis. Property preserving symmetric encryption. In *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 375–391, 2012.

[PTC18] Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock knock, who's there? membership inference on aggregate location data. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2018.

[PWLP19] Rishabh Poddar, Stephanie Wang, Jianan Lu, and Raluca Ada Popa. Practical volume-based attacks on encrypted databases. Cryptology ePrint Archive, Paper 2019/1224, 2019. `https://eprint.iacr.org/2019/1224`.

[Qdr] Qdrant. Collections. `https://qdrant.tech/documentation/concepts/collections/`. Accessed August 15, 2026.

[RG19] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[RKH+21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

[SB15] Suyash S. Shringarpure and Carlos D. Bustamante. Privacy risks from genomic data-sharing beacons. *The American Journal of Human Genetics*, 97(5):631–646, 2015.

[SDS+19a] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 5558–5567. PMLR, 2019.

[SDS+19b] Suhas Jayaram Subramanya, Fnu Devvrit, Harsha Vardhan Simhadri, Ravishankar Krishnaswamy, and Rohan Kadekodi. DiskANN: Fast accurate billion-point nearest neighbor search on a single node. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 13766–13776, 2019.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[SOJH09] Sriram Sankararaman, Guillaume Obozinski, Michael I. Jordan, and Eran Halperin. Genomic privacy and limits of individual detection in a pool. *Nature Genetics*, 41(9):965–967, 2009.

[SOT22] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data – anonymisation groundhog day. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 1451–1468, Boston, MA, 2022. USENIX Association.

[SR20] Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390, 2020.

[SSSS17] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[TRR+21] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[Wea] Weaviate. Vector search. `https://docs.weaviate.io/weaviate/concepts/search/vector-search`. Accessed August 15, 2026.

[WGCS22] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *International Conference on Learning Representations (ICLR)*, 2022.

[WHL+25] Guangshuo Wang, Jiajun He, Hao Li, Min Zhang, and Dengguo Feng. Rag-leaks: difficulty-calibrated membership inference attacks on retrieval-augmented generation. *Science China Information Sciences*, 68(6):160102, 2025.

[WLL+20] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550. Association for Computational Linguistics, 2020.

[YCWW15] Xingliang Yuan, Helei Cui, Xinyu Wang, and Cong Wang. Enabling privacy-assured similarity retrieval over millions of encrypted records. In *European Symposium on Research in Computer Security (ESORICS)*, pages 40–60, 2015.

[YGFJ18] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium (CSF)*, 2018.

[YGLL26] Huanyi Ye, Jiale Guo, Ziyao Liu, and Kwok-Yan Lam. Efficient privacy-preserving retrieval augmented generation with distance-preserving encryption. arXiv preprint arXiv:2601.12331, 2026. `https://arxiv.org/abs/2601.12331`.

[ZKP16] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 707–720. USENIX Association, 2016.

[ZPZP25] Jinhao Zhu, Liana Patel, Matei Zaharia, and Raluca Ada Popa. Compass: Encrypted semantic search with high accuracy. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 915–938, 2025.

[ZSF25] Mingxun Zhou, Elaine Shi, and Giulia Fanti. PACMANN: Efficient private approximate nearest neighbor search. In *International Conference on Learning Representations (ICLR)*, 2025.

[ZZH+24] Shenglai Zeng, Jiankun Zhang, Pengfei He, Yue Xing, Yiding Liu, Han Xu, Jie Ren, Shuaiqiang Wang, Dawei Yin, Yi Chang, and Jiliang Tang. The good and the bad: Exploring privacy issues in retrieval-augmented generation (rag). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4505–4524. Association for Computational Linguistics, 2024.

A Proofs for the RAP Security Results

This appendix proves the RAP results from Section 6. We keep the proofs modular, but the organization follows the argument rather than introducing separate terminology for each technical step.

Proof of Lemma 6.2. Recall that two leakage profiles are equivalent when each can be simulated from the other. We give simulations in both directions.

Given RAP hash values $h_1, \ldots, h_n$, output their Gram matrix. Write

$$h_i = Mx_i + \eta_i = M(x_i + \eta_i'), \qquad \eta_i' = M^{-1}\eta_i.$$

For every fixed M, the vectors $\eta_1', \ldots, \eta_n'$ are still independent and uniform in $\mathcal{B}(0; \beta/4)$ because the noise distribution is rotation-invariant. Since this conditional distribution does not depend on M, the transformed noise tuple is also independent of M. Orthogonality of M therefore gives

$$\Gamma(h_1, \ldots, h_n) = \Gamma(x_1 + \eta_1', \ldots, x_n + \eta_n'),$$

which has exactly the noisy-Gram distribution.

Conversely, suppose the simulator receives a positive semidefinite matrix $G \in \mathbb{R}^{n\times n}$ of rank at most d. To give a concrete factor, use the spectral theorem to write

$$G = U\Lambda U^\top$$

with U orthogonal and Λ diagonal. Let $r = \operatorname{rank}(G)$, let U_r contain the eigenvectors corresponding to the positive eigenvalues, and let Λ_r be the corresponding diagonal matrix. Set

$$Y = \begin{pmatrix} \Lambda_r^{1/2} U_r^\top \\ 0_{(d-r)\times n} \end{pmatrix}.$$

Because G is positive semidefinite, all its eigenvalues are nonnegative, and

$$Y^\top Y = U_r \Lambda_r U_r^\top = G.$$

Thus Y is an efficiently computable Gram factor. This spectral construction is only one convenient choice; the remainder of the proof uses only $Y^\top Y = G$. The simulator samples a fresh uniform $Q \leftarrow\!\!\$\ O(d)$ and returns the columns of QY.

To check the distribution, let $Z \in \mathbb{R}^{d\times n}$ be the actual noisy plaintext matrix whose Gram matrix is G, i.e., $G = Z^\top Z$. Since $Y^\top Y = Z^\top Z$, there is an orthogonal matrix $R \in O(d)$ with

$$Z = RY.$$

For completeness, define the map $y_i \mapsto z_i$ on the span of the columns of Y. Equality of Gram matrices implies that this map is well-defined and preserves inner products. It is therefore an isometry between the two column spans and extends to an orthogonal map on all of $\mathbb{R}^d$.

A real RAP output can be written $MZ = MRY$, where M is uniform and independent of Z. Conditional on Z, the matrix R is fixed, so MR is again uniform on $O(d)$. Hence MRY has the same distribution as QY. □

A.1 Proof of Theorem 6.5

We now prove the matched known-plaintext reduction. After a common orthogonal change of coordinates, the k known plaintexts lie in the first k coordinates. A sketch of an unknown noisy row records its projection onto these coordinates and its norm. Its component in $E^\perp$ therefore has a known length but an unknown direction. The simulator samples the noise on the known rows itself, so reconstructing the full Gram matrix reduces to sampling these unknown directions with the correct joint distribution.

Let

$$E = \mathrm{span}(e_1, \ldots, e_k).$$

Write $E^\perp$ for the orthogonal complement of E in $\mathbb{R}^d$. An orthogonal map of $\mathbb{R}^d$ that fixes every vector in E acts only on $E^\perp$, and we call such a map a *rotation of* $E^\perp$.

Lemma A.1. *Let Z be a random vector whose distribution is unchanged by every rotation of $E^\perp$. Then, conditional on*

$$P_E Z = t, \qquad \|Z\| = r,$$

the component $P_{E^\perp} Z$ is uniform on the sphere in $E^\perp$ of radius $\sqrt{r^2 - \|t\|^2}$, for almost every (t, r).

Proof. Fix one such rotation Q. The observation $(P_E Z, \|Z\|)$ is unchanged when Z is replaced by QZ, while QZ has the same unconditional distribution as Z. Hence

$$(QZ, P_E Z, \|Z\|) \equiv (Z, P_E Z, \|Z\|).$$

Conditioning on $(P_E Z, \|Z\|) = (t, r)$ therefore leaves the conditional distribution unchanged by every rotation of $E^\perp$, for almost every (t, r). On this conditioning event the component in $E^\perp$ has fixed length $\sqrt{r^2 - \|t\|^2}$. The unique probability distribution on that sphere that is unchanged by all orthogonal transformations of $E^\perp$ is the uniform distribution. If the radius is zero, the sphere is the singleton $\{0\}$. □

The planted row requires one additional observation. Its component in $E^\perp$ can stay correlated with the challenge after the sketch is revealed. We will not try to reproduce that row itself; we only need to reproduce the Gram matrix.

Lemma A.2. *Let W be a random tuple whose joint distribution is unchanged by every rotation of $E^\perp$, and let Z be independent of W. Let Q be an independent uniform rotation of $E^\perp$. Then*

$$\Gamma(QW, QZ) = \Gamma(W, Z)$$

for every value of Q, W and Z, not only in distribution. Moreover, QW has the same distribution as W, QW is independent of QZ, and, conditional on $(P_E Z, \|Z\|)$, the component $P_{E^\perp}(QZ)$ is uniform on the sphere of the corresponding radius.

Proof. The Gram matrices are equal because the same orthogonal map is applied to every vector.

For the distribution of QW, condition on $Q = q$. By assumption, qW has the same distribution as W for every fixed q, so averaging over Q leaves the distribution unchanged.

For independence, let A be any measurable set of tuples. Conditional on (Q, Z),

$$\Pr[QW \in A \mid Q, Z] = \Pr[W \in Q^{-1}A] = \Pr[W \in A].$$

The final quantity is independent of both Q and Z. Thus QW is independent of every function of (Q, Z), in particular of QZ.

Finally, condition on $Z = z$. The map Q fixes $P_E z$ and sends $P_{E^\perp} z$ uniformly over the sphere in $E^\perp$ having the same length. Averaging over all z with the same $(P_E Z, \|Z\|)$ proves the last claim. □

Proof of Theorem 6.5. Let $x_1, \dots, x_k$ be the known plaintexts sampled by the reduction. Since $k < d$ and these vectors are independent and uniform on the sphere, they are linearly independent with probability one. Choose an orthogonal matrix U such that

$$a_j := U^\top x_j \in E \qquad (j = 1, \dots, k).$$

Applying $U^\top$ to every noisy database row leaves every Gram entry unchanged and preserves the noise distribution. We may therefore work in these coordinates.

The reduction receives a challenge q from the planted-sketch problem and sets $p = Uq$. Conditional on the known plaintexts, U is fixed and q is a fresh uniform point on the sphere. Therefore p is uniform on the sphere and independent of the known plaintexts, exactly as in the membership game.

For each known plaintext the reduction samples fresh noise η_j and uses

$$z_j = a_j + \eta_j.$$

These have exactly the distribution of the known noisy rows after the change of coordinates.

Consider an unknown row with sketch

$$(t, r) = (P_E Z, \|Z\|).$$

If the row is not planted, then $Z = X + \eta$ for an independent uniform $X \in S^{d-1}$ and independent uniform ball noise η. The distribution of Z is unchanged by every orthogonal transformation, so Lemma A.1 applies. Conditional on the sketch, the row is therefore distributed as

$$t + \sqrt{r^2 - \|t\|^2}\, V,$$

where V is uniform on the unit sphere of $E^\perp$. This is exactly how the reduction reconstructs a nonplanted row.

Different unknown rows are independent before their sketches are revealed, and each sketch depends only on its own row. Conditioning on all N sketches therefore keeps the rows independent, and row i has the conditional distribution just described given its own sketch $S_i = s_i$. This holds for almost every value of the sketch tuple. The reduction may therefore reconstruct each nonplanted row on its own. Appendix A.2 writes out both of these conditioning facts in full.

Suppose now that one row is planted. Write it as

$$Z_\star = q + \eta_\star.$$

Its remaining direction can stay correlated with q after conditioning on its sketch, so replacing that direction independently would not reproduce the row itself.

Let W denote all other rows after the same coordinate change and after conditioning on all unknown-row sketches. The product identity above shows that W remains independent of $Z_\star$. Its joint conditional distribution is unchanged by every rotation of $E^\perp$. Each known plaintext lies in E and has rotation-invariant noise, while every nonplanted unknown row has the conditional distribution just described.

Apply one fresh uniform rotation Q of $E^\perp$ to every row. By Lemma A.2, no Gram entry changes. The other rows keep the same joint distribution and remain independent of the planted row, while the planted row's component in $E^\perp$ becomes uniform conditional on its sketch. Consequently the Gram matrix has exactly the same conditional distribution as if the planted row had been reconstructed from its sketch with a fresh uniform remaining direction. This is precisely what the reduction does.

We have shown that, conditional on the known plaintexts, the challenge, and all sketches, the reduction produces exactly the Gram-matrix distribution of the matched membership game under the same challenge bit. The planted-sketch game chooses the planted unknown row uniformly among its $N = n - k$ positions. The reduction applies the same independent uniform relabeling used in the matched game and reports the corresponding positions of the known plaintexts. Hence the complete views are identically distributed for each challenge bit.

The reduction returns the adversary's output unchanged, so the two advantages are equal. Apart from one call to the adversary, the reduction uses only sampling and polynomial-time linear algebra. □

Proof of Theorem 6.4. Given the noisy Gram view, use Lemma 6.2 to simulate the RAP hash values and run the RAP adversary. The simulation is exact, so the resulting Gram-view adversary has the same advantage. Theorem 6.5 then gives a planted-sketch distinguisher with that same advantage, and Theorem 6.6 bounds every such distinguisher by $\frac{C}{\sqrt{N}}(c/\beta)^{k/2}$ with $N = n - k$. □

A.2 Conditioning Step in the Proof of Theorem 6.5

The proof of Theorem 6.5 conditions on one sketch from each unknown row. Two facts are needed after this conditioning: rotations of the coordinates outside E must remain symmetries of each ordinary row, and rows that were independent before conditioning must remain independent. Both facts follow directly from the way the sketches are defined.

Let W be a random vector or tuple whose distribution is unchanged by every orthogonal map Q that fixes E. Let $S = S(W)$ satisfy $S(QW) = S(W)$ for every such Q. In our application,

$$S(W) = (P_E W, \|W\|).$$

Fix Q. For every measurable set A of possible values of W and every measurable set B of possible sketch values,

$$\begin{aligned}\Pr[QW \in A, S \in B] &= \Pr[QW \in A, S(QW) \in B] \\ &= \Pr[W \in A, S(W) \in B].\end{aligned}$$

The first equality uses $S(QW) = S(W)$ pointwise, and the second uses $QW \equiv W$. Thus (QW, S) and (W, S) have the same joint distribution. Conditioning on $S = s$ gives, for almost every s,

$$\Pr[QW \in A \mid S = s] = \Pr[W \in A \mid S = s].$$

So revealing the sketch does not select a preferred direction outside E.

Now let $Z_1, \ldots, Z_N$ be independent rows and let $S_i = S_i(Z_i)$ be the sketch of row i. Each sketch depends on only one row. The joint conditional distribution of the rows given all N sketches therefore factors into the individual conditional distributions, for almost every value of the sketch tuple. Conditioning on all sketches thus keeps the rows independent, and row i has the conditional distribution of Z_i given its own sketch $S_i = s_i$. In particular, conditioning on all sketches does not create dependence between the planted row and the remaining rows.

These are exactly the two facts used by the common-rotation argument. The ordinary rows remain symmetric in the unseen coordinates, and the planted row remains independent of them. Applying one common random rotation to all rows therefore leaves the Gram matrix unchanged while randomizing the planted row's unseen direction.

A.3 Proof of Theorem 6.6

The proof separates two effects. First, the planted sketch is hidden among N positions. This gives the factor $N^{-1/2}$ exactly. Second, one sketch reveals only k coordinates and a norm. We show that the information in one sketch grows exponentially in k but has no exponential dependence on the remaining dimension $m = d - k$.

Lemma A.3. *Fix $q \in S^{d-1}$. Let μ be the distribution of a background sketch and ν_q the distribution of a planted sketch. Let $P_{0,q} = \mu^N$, and let $P_{1,q}$ be the distribution obtained by replacing one uniformly random factor of μ^N by ν_q. Then*

$$\chi^2(P_{1,q} \| P_{0,q}) = \frac{1}{N} \chi^2(\nu_q \| \mu).$$

Consequently,

$$\delta_{d,k,N,\beta} \le \frac{1}{2\sqrt{N}} \mathbb{E}_{q \leftarrow \$ S^{d-1}} \sqrt{\chi^2(\nu_q \| \mu)}.$$

Proof. Let $\ell_q = d\nu_q / d\mu$ be the ratio of the densities of ν_q and μ. The likelihood ratio of $P_{1,q}$ against $P_{0,q}$, that is the density of $P_{1,q}$ relative to $P_{0,q}$, is then

$$L_q(s_1, \dots, s_N) = \frac{1}{N} \sum_{i=1}^{N} \ell_q(s_i).$$

The variables $\ell_q(s_i)$ are independent and each has mean one. Therefore the cross terms vanish in the expansion of $\mathbb{E}[(L_q - 1)^2]$, giving

$$\chi^2(P_{1,q} \| P_{0,q}) = \frac{1}{N} \chi^2(\nu_q \| \mu).$$

Statistical distance is at most one half the square root of chi-square divergence. This is Cauchy–Schwarz in one line,

$$\begin{aligned} \Delta(P, Q) &= \tfrac{1}{2} \, \mathbb{E}_Q \left| \frac{dP}{dQ} - 1 \right| \\ &\le \tfrac{1}{2} \sqrt{\mathbb{E}_Q \left(\frac{dP}{dQ} - 1 \right)^2} \\ &= \tfrac{1}{2} \sqrt{\chi^2(P \| Q)}. \end{aligned}$$

Since the challenge q has the same marginal distribution under both bits, averaging the conditional bound over q gives the second claim. □

We first record a crude bound that is useful when the challenge happens to lie very close to the revealed k-dimensional subspace.

Proposition A.4. *Let $0 < \beta < 4$, set $a = \beta/4$, and let $N \ge 1$. Then*

$$\delta_{d,k,N,\beta} \le \min \left\{ 1, \frac{1}{2\sqrt{N}} \left(\frac{(1+a)^d - (1-a)^d}{a^d} - 1 \right)^{1/2} \right\}.$$

Proof. Fix $q \in S^{d-1}$ and write $Z = q + \eta$, where η is uniform in the ball of radius a. Let $R = \|Z\|$ and $V = Z/R$. Conditional on $R = r$, the direction V is uniform on the spherical cap

$$C_r(q) = \left\{ v \in S^{d-1} : \langle v, q \rangle \geq \frac{r^2 + 1 - a^2}{2r} \right\}.$$

To see this, expand $\|Z - q\|^2 = \|Z\|^2 - 2\langle Z, q \rangle + 1$. Given $\|Z\| = r$, the constraint $\|Z - q\| \leq a$ is therefore the same as $\langle V, q \rangle \geq (r^2 + 1 - a^2)/(2r)$. The density of η is constant on the ball, so Z is uniform on the ball, and its conditional distribution on the sphere of radius r is uniform on the part of that sphere the constraint leaves, which is exactly $C_r(q)$. For a background row, conditional on the same radius, the direction is uniform on the whole sphere. Let $p_a(r)$ be the normalized area of this cap. Before projecting to the sketch, the conditional likelihood ratio is $1/p_a(r)$ on the cap and zero elsewhere, so its chi-square divergence is $1/p_a(r) - 1$. Data processing can only decrease chi-square divergence.

The radial density of a point uniform in $\mathcal{B}(q; a)$ is

$$f_R(r) = \frac{d}{a^d} r^{d-1} p_a(r), \qquad 1 - a \leq r \leq 1 + a.$$

Therefore

$$\begin{aligned} 1 + \chi^2(\nu_q \| \mu) &\leq \mathbb{E}_R \frac{1}{p_a(R)} \\ &= \frac{d}{a^d} \int_{1-a}^{1+a} r^{d-1} \, dr \\ &= \frac{(1+a)^d - (1-a)^d}{a^d}. \end{aligned}$$

The proposition follows from Lemma A.3. □

For $j \geq 2$ and $u \in (0, 1)$, write

$$\overline{F}_j(u) = \Pr_{V \leftarrow \$ S^{j-1}}[\langle V, e_1 \rangle \geq u].$$

This is the normalized area of the spherical cap of height u on S^{j-1}, that is the chance that a uniformly random direction in j dimensions falls inside that cap. Every cap probability below is expressed through it.

Lemma A.5. *Let $d = k + m$ with $m \geq 2$, fix $q \in S^{d-1}$, and write $t = \|P_E q\|$. For $\tau \in (t, 1)$, let $\lambda_{\tau,q}$ be the distribution of $P_E V$ when V is uniform on*

$$C_\tau(q) = \{ v \in S^{d-1} : \langle v, q \rangle \geq \tau \},$$

and let λ_0 be the distribution of $P_E V$ for uniform $V \in S^{d-1}$. Then

$$1 + \chi^2(\lambda_{\tau,q} \| \lambda_0) \leq \frac{\overline{F}_m(c)}{\overline{F}_d(\tau)}, \qquad c = \sqrt{\frac{\tau^2 - t^2}{1 - t^2}}.$$

Moreover, for an absolute constant C,

$$\overline{F}_m(c) \leq \frac{C}{\sqrt{m}} (1 - t^2)^{-(m-2)/2} (\tau^2 - t^2)^{-1/2} (1 - \tau^2)^{(m-1)/2}.$$

Proof. Write $q_E = P_E q$ and $q_\perp = q - q_E$. Conditional on $Y = P_E V = y$, the remaining part of a uniform $V \in S^{d-1}$ is

$$\sqrt{1 - \|y\|^2}\, W, \qquad W \leftarrow\!\!\$\; S^{m-1} \subseteq E^\perp.$$

Thus the conditional probability of the cap event is

$$h(y) = \overline{F}_m\left(\frac{\tau - \langle y, q_E\rangle}{\sqrt{1-t^2}\sqrt{1-\|y\|^2}}\right).$$

Since the cap has normalized area $\overline{F}_d(\tau)$, the likelihood ratio of $\lambda_{\tau,q}$ with respect to λ_0 is $L(y) = h(y)/\overline{F}_d(\tau)$. Because $\mathbb{E}L = 1$,

$$1 + \chi^2(\lambda_{\tau,q}\|\lambda_0) = \mathbb{E}L^2 \le \|L\|_\infty.$$

For fixed $\|y\| = r$, the argument of $\overline{F}_m$ is minimized when y is parallel to q_E. We therefore minimize

$$\frac{\tau - tr}{\sqrt{1-t^2}\sqrt{1-r^2}}.$$

A derivative gives the unique minimum at $r = t/\tau$, with value $c = \sqrt{(\tau^2 - t^2)/(1-t^2)}$. Hence $\|h\|_\infty = \overline{F}_m(c)$, proving the first claim.

The first coordinate of a uniform point on S^{m-1} has density

$$\kappa_m(1-u^2)^{(m-3)/2}, \qquad \kappa_m = \frac{\Gamma(m/2)}{\sqrt{\pi}\,\Gamma((m-1)/2)}.$$

For $c \in (0,1)$, direct integration gives

$$\overline{F}_m(c) \le \frac{\kappa_m}{(m-1)c}(1-c^2)^{(m-1)/2}.$$

Using $\kappa_m \le C\sqrt{m}$ together with

$$1 - c^2 = \frac{1-\tau^2}{1-t^2}, \qquad c^{-1} = \sqrt{\frac{1-t^2}{\tau^2 - t^2}}$$

gives the second claim. □

The last analytic ingredient averages the fixed-challenge bound over the random amount of the challenge lying in E. The proof is a one-dimensional Beta integral; the Gamma estimates used to make the cancellation of $m = d - k$ explicit are in Appendix A.4.

Lemma A.6. *Let $m, k \ge 1$, set $d = m + k$, let $0 < a \le 1/2$, and set $b = \sqrt{1-a^2}$. If*

$$T^2 \sim \mathrm{Beta}(k/2, m/2),$$

then there is an absolute constant C such that, for $m \ge 2$,

$$\sqrt{\frac{d}{m}}\,\mathbb{E}\Big[(1-T^2)^{-(m-2)/4}(b^2 - T^2)^{-1/4}\mathbf{1}[T < b]\Big] \le C^k,$$

and, for every $m \ge 1$,

$$\Pr[T \ge b]\left(\frac{(1+a)^d - (1-a)^d}{a^d}\right)^{1/2} \le C^k a^{-k/2}.$$

Proof. Let $U = 1 - T^2$, so $U \sim \mathrm{Beta}(m/2, k/2)$. Ignoring the leading factor $\sqrt{d/m}$ for the moment, the first expectation equals

$$\frac{1}{B(m/2,k/2)} \int_{a^2}^1 u^{(m-2)/4}(u-a^2)^{-1/4}(1-u)^{k/2-1}\,du.$$

With $u = a^2 + b^2 v$, this becomes

$$\frac{b^{k-1/2}}{B(m/2,k/2)} \int_0^1 (a^2+b^2v)^{(m-2)/4} v^{-1/4}(1-v)^{k/2-1}\,dv.$$

If $k > m$, discard the factor $(a^2+b^2v)^{(m-2)/4} \le 1$. The resulting ratio is

$$\frac{B(3/4,k/2)}{B(m/2,k/2)} = \frac{\Gamma(3/4)\Gamma((m+k)/2)}{\Gamma(m/2)\Gamma(k/2+3/4)} \le C^k.$$

The final inequality follows directly from Stirling and is uniform over $m < k$.

Suppose now that $k \le m$. For $m = 2$ the same Beta-function bound suffices. For $m \ge 3$, since $b^2 \ge 3/4$,

$$(a^2+b^2v)^{(m-2)/4} \le e^{-cm(1-v)}$$

for an absolute $c > 0$. Split the integral at $v = 1/2$. On $v \le 1/2$ it is at most Ce^{-cm}. On $v \ge 1/2$, set $w = 1 - v$ and extend the integral to $[0, \infty)$ to obtain

$$C \int_0^\infty e^{-cmw} w^{k/2-1}\,dw = Cc^{-k/2}\Gamma(k/2)m^{-k/2}.$$

After dividing by $B(m/2, k/2)$, the two required estimates are

$$\frac{\Gamma((m+k)/2)}{\Gamma(m/2)} m^{-k/2} \le C^k$$

and

$$e^{-cm}\frac{\Gamma((m+k)/2)}{\Gamma(m/2)\Gamma(k/2)} \le C^k.$$

Both follow from Stirling uniformly for $k \le m$; the calculations are written out in Appendix A.4. Finally, $\sqrt{d/m} \le \sqrt{k+1} \le 2^{k/2}$, proving the first estimate.

For the tail, on $0 \le u \le a^2 \le 1/4$ the Beta density gives

$$\Pr[T \ge b] = \Pr[U \le a^2] \le C\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)} a^m.$$

Also,

$$(1+a)^d - (1-a)^d \le 2da(1+a)^{d-1}.$$

Combining the two bounds and setting $\theta = \sqrt{a(1+a)} < 1$ gives

$$\Pr[T \ge b]\left(\frac{(1+a)^d - (1-a)^d}{a^d}\right)^{1/2} \le C^k a^{-k/2}\sqrt{d}\,\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)}\theta^{m-1}.$$

Here factors depending only exponentially on k, including $(1+a)^{k/2}$, have been absorbed into C^k; Appendix A.4 keeps these factors explicit before absorbing them.

If $k \le m$, Stirling reduces the remaining m-dependence to

$$C^k \left(\frac{m}{k}\right)^{(k-1)/2} \theta^{m-1}.$$

The logarithm of the factor after C^k is

$$\frac{k-1}{2}\log(m/k) + (m-1)\log\theta,$$

which is concave in m and has maximum $O(k)$. Thus the factor is at most C^k. If $k > m$, the entropy form of Stirling gives the Gamma ratio at most $C^{m+k} \le C^{2k}$, while $\sqrt{d} \le C^k$ and $\theta^{m-1} \le 1$. This proves the tail estimate. □

Proof of Theorem 6.6. Set

$$a = \beta/4, \qquad m = d-k, \qquad b = \sqrt{1-a^2}.$$

Fix a challenge q and put $t = \|P_E q\|$.

For a planted noisy row $Z = q + \eta$, let $R = \|Z\|$ and $V = Z/R$. Conditional on $R = r$, the direction V is uniform on the cap

$$C_{\tau(r)}(q), \qquad \tau(r) = \frac{r^2 + 1 - a^2}{2r}.$$

The background direction conditional on the same radius is uniform on the whole sphere. The two distributions of the radius are the same. Moreover,

$$\min_{1-a \le r \le 1+a} \tau(r) = b.$$

The minimum is at $r = b$ by the arithmetic–geometric mean inequality applied to $\tau(r) = \frac{1}{2}\big(r + (1-a^2)/r\big)$.

Assume first that $m \ge 2$ and $t < b$. Lemma A.5, averaged over the common radius, gives

$$1 + \chi^2(\nu_q \| \mu) \le \frac{Cd}{a^d\sqrt{m}} (1-t^2)^{-(m-2)/2} (b^2 - t^2)^{-1/2} \cdot \int_{1-a}^{1+a} r^{d-1} (1 - \tau(r)^2)^{(m-1)/2}\, dr.$$

The cap area disappears here because its factor in the conditional divergence cancels the same factor in the radial density.

The remaining radial integral is elementary. Define

$$\rho(r) = r\sqrt{1-\tau(r)^2} = \frac{1}{2}\sqrt{((1+a)^2 - r^2)(r^2 - (1-a)^2)}.$$

Since $d = k + m$, the integrand equals $r^k \rho(r)^{m-1}$. With

$$u = \frac{r^2 - (1-a)^2}{4a}$$

we have $\rho(r) = 2a\sqrt{u(1-u)}$ and $dr = 2a\,du/r$. Therefore

$$\begin{aligned}\int_{1-a}^{1+a} & r^{d-1}(1-\tau(r)^2)^{(m-1)/2}\,dr \\ &= (2a)^m \int_0^1 r(u)^{k-1}[u(1-u)]^{(m-1)/2}\,du \\ &\le C^k a^m m^{-1/2},\end{aligned}$$

where we used $r(u) \le 1 + a \le 3/2$ and the standard central-Beta estimate

$$B((m+1)/2, (m+1)/2) \le C2^{-m}m^{-1/2}.$$

Hence

$$1 + \chi^2(\nu_q\|\mu) \le C^k \frac{d}{m} a^{-k}(1-t^2)^{-(m-2)/2}(b^2-t^2)^{-1/2}.$$

For uniform q,

$$T^2 := \|P_E q\|^2 \sim \mathrm{Beta}(k/2, m/2).$$

Taking square roots, averaging over $T < b$, and applying the first part of Lemma A.6 gives

$$\mathbb{E}_q\left[\sqrt{\chi^2(\nu_q\|\mu)}\mathbf{1}[T < b]\right] \le C^k a^{-k/2}.$$

On the event $T \ge b$, use the crude fixed-q bound from Proposition A.4:

$$1 + \chi^2(\nu_q\|\mu) \le \frac{(1+a)^d - (1-a)^d}{a^d}.$$

The second part of Lemma A.6 gives

$$\mathbb{E}_q\left[\sqrt{\chi^2(\nu_q\|\mu)}\mathbf{1}[T \ge b]\right] \le C^k a^{-k/2}.$$

Combining the two events proves

$$\mathbb{E}_q\sqrt{\chi^2(\nu_q\|\mu)} \le C^k a^{-k/2}$$

when $m \ge 2$.

If $m = 1$, the crude bound alone suffices because $d = k + 1$. The mean-value estimate

$$(1+a)^d - (1-a)^d \le 2da(1+a)^{d-1}$$

gives

$$\left(\frac{(1+a)^d - (1-a)^d}{a^d}\right)^{1/2} \le C^k a^{-k/2}.$$

Thus the same single-sketch estimate holds for every $m = d - k \ge 1$.

Finally $a = \beta/4$. After changing absolute constants,

$$\mathbb{E}_q\sqrt{\chi^2(\nu_q\|\mu)} \le C\left(\frac{c}{\beta}\right)^{k/2}.$$

Combining this with Lemma A.3 gives

$$\delta_{d,k,N,\beta} \le \frac{C}{\sqrt{N}}\left(\frac{c}{\beta}\right)^{k/2},$$

which proves the theorem. □

A.4 Gamma Bounds Used in Lemma A.6

The proof of Lemma A.6 uses a few estimates involving Gamma functions. Their only purpose is to show that the bounds depend exponentially on the number k of known plaintexts, but not on the remaining dimension $m = d - k$. We give the calculations here because this dimension independence is important to Theorem 6.6.

We use Stirling's estimate in the following standard form. There are absolute constants $c_0, C_0 > 0$ such that, for every $z \geq 1/2$,

$$c_0 z^{z-1/2} e^{-z} \leq \Gamma(z) \leq C_0 z^{z-1/2} e^{-z}.$$

As usual, $C > 1$ below denotes an absolute constant whose value may change from one occurrence to the next.

We first need, when $2 \leq m < k$,

$$\frac{\Gamma(3/4)\Gamma((m+k)/2)}{\Gamma(m/2)\Gamma(k/2+3/4)} \leq C^k. \tag{7}$$

Set $x = m/2$ and $y = k/2$. Applying Stirling to the three Gamma factors that depend on m or k, and absorbing fixed powers and constants, gives

$$\log \frac{\Gamma(x+y)}{\Gamma(x)\Gamma(y+3/4)} \leq (x+y)\log(x+y) \\ - x\log x - y\log y + O(x+y).$$

For positive x, y, moreover,

$$(x+y)\log(x+y) - x\log x - y\log y \\ \leq (x+y)\log 2.$$

Since $m < k$, we have $x + y < k$. The right-hand side is therefore $O(k)$, which proves (7). In particular, the bound is uniform in m.

Next suppose $1 \leq k \leq m$. We use

$$\frac{\Gamma((m+k)/2)}{\Gamma(m/2)} m^{-k/2} \leq C^k. \tag{8}$$

Put again $x = m/2$ and now $h = k/2$, so $0 < h \leq x$. Stirling gives

$$\log \frac{\Gamma(x+h)}{\Gamma(x)} \leq h\log(x+h) + O(h).$$

Because $h \leq x$, we have $x + h \leq 2x = m$. Subtracting $h \log m$ therefore leaves only $O(h) = O(k)$, proving (8).

In the same range $1 \leq k \leq m$, dividing by the Beta normalizing constant produces the factor

$$e^{-cm} \frac{\Gamma((m+k)/2)}{\Gamma(m/2)\Gamma(k/2)}, \tag{9}$$

where $c > 0$ is an absolute constant. We show that it is at most C^k without hiding any m-dependence. Stirling gives

$$\frac{\Gamma((m+k)/2)}{\Gamma(m/2)\Gamma(k/2)} \leq C^k \left(\frac{m}{k}\right)^{k/2}.$$

Here and below a factor of the form e^{Ck} is absorbed into C^k. Writing $u = m/k \geq 1$, multiplication by e^{-cm} leaves

$$\exp\left(k\left[-cu + \frac{1}{2}\log u\right]\right).$$

The function $-cu + \frac{1}{2}\log u$ has a finite maximum on $[1, \infty)$. Hence the factor in (9) is at most C^k.

Finally, the tail estimate uses, when $k > m \geq 1$,

$$\frac{\Gamma((m+k)/2)}{\Gamma(m/2+1)\Gamma(k/2)} \leq C^k. \tag{10}$$

A direct application of Stirling gives

$$\log \frac{\Gamma((m+k)/2)}{\Gamma(m/2+1)\Gamma(k/2)} \leq O(m+k).$$

Since $m < k$, we have $m + k < 2k$, so the last quantity is $O(k)$ and (10) follows.

For completeness, we spell out the one place in the tail calculation where several elementary factors are combined. From the Beta density one obtains

$$\Pr[T \geq b] \leq C\,\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)}a^m, \qquad d = m + k,$$

while the mean-value theorem gives

$$\left(\frac{(1+a)^d - (1-a)^d}{a^d}\right)^{1/2} \leq C\sqrt{d}\,(1+a)^{(d-1)/2}a^{(1-d)/2}.$$

Multiplying these inequalities gives

$$C\sqrt{d}\,\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)}a^{(m+1-k)/2}(1+a)^{(m+k-1)/2}.$$

Let

$$\theta = \sqrt{a(1+a)}.$$

Because $0 < a \leq 1/2$, we have $\theta \leq \sqrt{3/4} < 1$. Also $(1+a)^{k/2} \leq (3/2)^{k/2}$ and $a \leq 1$. Thus the previous display is at most

$$C^k a^{-k/2}\sqrt{d}\,\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)}\theta^{m-1}. \tag{11}$$

This is the transition that is easy to lose track of if the factors are absorbed implicitly.

It remains to show that the part of (11) following $a^{-k/2}$ is at most C^k. If $k > m$, this follows immediately from (10), together with $\sqrt{d} \leq C^k$ and $\theta^{m-1} \leq 1$.

Suppose instead that $k \leq m$. Stirling gives

$$\frac{\Gamma(d/2)}{\Gamma(m/2+1)\Gamma(k/2)} \leq C^k\frac{m^{k/2-1}}{\Gamma(k/2)}.$$

Since $d = m + k \leq 2m$, multiplying by $\sqrt{d}$ and applying the lower Stirling bound to $\Gamma(k/2)$ reduces the remaining m-dependence to

$$C^k\left(\frac{m}{k}\right)^{(k-1)/2}\theta^{m-1}.$$

		bound on matched MIA advantage		
k	$\Xi(384, k, 0.5)$	$n = 10^4$	$n = 10^5$	$n = 10^6$
1	3.20	0.016	0.005	0.002
2	11.31	0.057	0.018	0.006
3	38.28	0.191	0.061	0.019
4	128.8	0.644	0.204	0.064
5	433.0	—	0.685	0.217
6	1454	—	—	0.727

Table 5: Estimates of $\Xi(d, k, \beta)$ at $d = 384$ and $\beta = 0.5$, and the bound $\Xi/(2\sqrt{n-k})$ they give on the matched MIA advantage against RAP. A dash marks an entry at least 1, where the bound says nothing.

The logarithm of the factor after C^k is

$$g(m) = \frac{k-1}{2} \log \frac{m}{k} + (m-1) \log \theta.$$

This is a concave function of m. Its unconstrained maximum occurs at

$$m_0 = \frac{k-1}{2 \log(1/\theta)}.$$

If $m_0 \le k$, then g is decreasing on $[k, \infty)$ and $g(m) \le g(k) \le 0$. If $m_0 > k$, then $\log(1/\theta) < 1/2$, while $\theta \le \sqrt{3/4}$ gives the absolute lower bound $\log(1/\theta) \ge \log(2/\sqrt{3})$. Hence m_0/k is bounded by an absolute constant. Dropping the nonpositive term $(m_0 - 1) \log \theta$ then gives $g(m_0) = O(k)$. Thus this factor is at most C^k in either case.

These are all of the Gamma-function estimates used in the proof of Lemma A.6. Each estimate is uniform over $m = d - k$, so no dependence exponential in the ambient dimension is absorbed into the constants.

A.5 Estimating the Bound Numerically

Theorem 6.6 leaves c and C unspecified, so on its own it gives no number at any particular (d, k, β). The quantity the theorem bounds,

$$\Xi(d, k, \beta) = \mathbb{E}_{q \leftarrow \$ S^{d-1}} \left[\sqrt{\chi^2(\nu_{q,d,k,\beta} \,\|\, \mu_{d,k,\beta})} \right],$$

is an explicit function of d, k and β, and we compute it by numerical quadrature. This uses nothing about c and C. What we report is a numerical calculation and not a proof, so we call it an estimate. By Lemma 6.2, Theorem 6.5 and Lemma A.3, the matched MIA advantage against RAP on a database of n vectors is at most $\Xi(d, k, \beta)/(2\sqrt{n-k})$.

Reduction to one radius integral. The sketch of a row Z is $(P_E Z, \|Z\|)$. Its norm $\|Z\|$ has the same distribution under both hypotheses, because the planted row is $q + \eta$ with $\|q\| = 1$, a background row is $X + \eta$ with $\|X\| = 1$, and the noise distribution is rotation-invariant. Conditioning on $\|Z\| = r$ therefore gives

$$\chi^2(\nu_q \| \mu) = \mathbb{E}_r \left[\chi^2\big(\lambda_{\tau(r),q} \,\|\, \lambda_0\big) \right],$$

where $\lambda_{\tau,q}$ is the distribution of $P_E V$ for V uniform on the cap $\{v \in S^{d-1} : \langle v, q \rangle \ge \tau\}$, λ_0 is the distribution of $P_E V$ for V uniform on S^{d-1}, and $\tau(r) = (1 + r^2 - a^2)/(2r)$ with $a = \beta/4$, as in the proof of Theorem 6.6.

The density ratio. The density of $\lambda_{\tau,q}$ relative to λ_0 has a closed form. Condition on $P_E V = t$. The component of V outside E then has length $\sqrt{1 - \|t\|^2}$ and a uniformly random direction, so

$$\langle V, q \rangle = \langle t, P_E q \rangle + \|P_{E^\perp} q\| \sqrt{1 - \|t\|^2}\, u,$$

where u is the first coordinate of a uniform point on S^{d-k-1}. Write

$$G(t) = \Pr[\, u \geq z(t) \,], \qquad z(t) = \frac{\tau - \langle t, P_E q \rangle}{\|P_{E^\perp} q\| \sqrt{1 - \|t\|^2}}.$$

Then $G(t)$ is the probability that V lies in the cap given $P_E V = t$. Hence $d\lambda_{\tau,q}/d\lambda_0 = G/\mathbb{E}_{\lambda_0}[G]$ and

$$1 + \chi^2(\lambda_{\tau,q} \| \lambda_0) = \frac{\mathbb{E}_{\lambda_0}[G^2]}{\big(\mathbb{E}_{\lambda_0}[G]\big)^2}.$$

Angular coordinates. Write $\|P_E q\| = \sin\alpha$ and $\|t\| = \sin\psi$ with $\alpha, \psi \in [0, \pi/2]$, and let w be the cosine of the angle between t and $P_E q$ inside E. In these coordinates

$$z = \frac{\tau - \sin\alpha \sin\psi\, w}{\cos\alpha \cos\psi},$$

so $z < 1$ for some w exactly when $\tau < \cos(\psi - \alpha)$. The function G is therefore zero outside the window $|\psi - \alpha| \leq \arccos\tau$, and only that window contributes to either expectation. The window is what makes the quadrature accurate in high dimension. At $d = 384$ and $\beta = 0.5$ it has width 0.25, against a full range of $\pi/2$ for ψ, so a grid of a few hundred points resolves the region where G is large. For uniform q and uniform V both $\sin^2\alpha$ and $\sin^2\psi$ have the $\mathrm{Beta}(k/2, (d-k)/2)$ distribution, and w is the first coordinate of a uniform point on S^{k-1}. We take Simpson's rule in all four variables α, r, ψ and w.

Evaluating the tail. In high dimension the cap is small and G underflows in double precision. At $d = 384$ and $\beta = 0.5$ the threshold is $\tau \approx 0.992$ and G is of order e^{-750}, so we work with $\log G$ throughout. We use

$$\begin{aligned} \log G = (p+1)\log(1 - z^2) - \log\big(2z(p+1)\big) \\ - \log B(\tfrac{1}{2}, p+1), \qquad p = \tfrac{d-k-3}{2}. \end{aligned}$$

This follows from

$$\int_z^1 (1 - u^2)^p \, du = \frac{(1 - z^2)^{p+1}}{2z(p+1)}\big(1 + O(1/p)\big).$$

Checks. For every α and τ, $\mathbb{E}_{\lambda_0}[G]$ equals $\Pr[\langle V, q \rangle \geq \tau]$ for V uniform on S^{d-1}, which has a closed form, so this identity tests the geometry and every normalization at once; weighted by the distributions the quadrature uses, the two sides agree to 3×10^{-7} in log at $k = 2$ and to 1×10^{-5} at $k = 3$. As an independent check on the final quantity, at fixed α and τ the quadrature gives $\chi^2 = 504.22$ for $d = 384$, $k = 3$, and $\beta = 0.5$, and direct sampling of t gives 506.4 from 10^7 samples, approaching the quadrature value from above as the sample size grows.

Values. We report Ξ and the resulting bound on the matched MIA advantage in Table 5. Two features are worth noting. The growth factor per known plaintext settles at about 3.36, which is the $(c/\beta)^{k/2}$ shape of Theorem 6.6 with c/β near 11.3. And Ξ barely moves with the dimension. At $k = 2$ and $\beta = 0.5$ it is 11.344 at $d = 128$, 11.314 at $d = 384$ and 11.303 at $d = 1536$, which is the dimension-freeness the theorem asserts for $\beta \leq 2$.

B Details for the RAP Attacks

This appendix gives the parts of Section 6 that the body states without derivation.

B.1 Normalized Gram entries are within β

Both matched and unmatched attacks compare inner products the adversary computes from plaintexts it holds against entries of the normalized matrix

$$\widehat{G}[i,j] = \frac{G[i,j]}{\sqrt{G[i,i]\,G[j,j]}},$$

which the adversary forms from G because the diagonal of G gives the squared norms of the vectors $x_i + \eta_i$. What matters is how close those entries are to the true inner products.

Claim B.1. For unit database vectors, $\left|\widehat{G}[i,j] - \langle x_i, x_j\rangle\right| \leq \beta$.

Proof. Write $v = x + \eta$ for the noisy version of a unit vector x and $\hat{v} = v/\|v\|$ for its normalization. Because $\|x\| = 1$, the norm $\|v\|$ differs from 1 by at most $\|\eta\|$, so normalizing moves v by at most $\|\eta\|$, and

$$\|\hat{v} - x\| \leq \|\hat{v} - v\| + \|v - x\| \leq 2\,\|\eta\| \leq \beta/2.$$

Now write

$$\langle \hat{v}_i, \hat{v}_j\rangle - \langle x_i, x_j\rangle = \langle \hat{v}_i - x_i, \hat{v}_j\rangle + \langle x_i, \hat{v}_j - x_j\rangle.$$

Both $\hat{v}_j$ and x_i are unit vectors, so each term is at most $\beta/2$ in absolute value and the difference is at most β. □

B.2 Matched Attack Measurements

We also run the matched attack on 512-dimensional FaceNet [SKP15] embeddings of the funneled LFW face images [HRBLM07]. We take one image per identity, producing 5,749 embeddings, so the experiment tests membership of a particular image's embedding rather than whether an identity is present through another photograph. Because every identity contributes exactly one embedding and each game samples without replacement, its database and nonmember are identity-disjoint. Table 6 contains the first tested breakpoint at two database sizes. At the preceding tested value of k, the advantages are 0.094 ± 0.030 and 0.160 ± 0.016 for $n = 100$ and $n = 1{,}000$, so each breakpoint is separated from the preceding point. FaceNet's VGGFace2 pretraining may include some LFW identities. Members and nonmembers are drawn from the same pool, so this cannot bias the membership decision, but it does mean these embeddings may not represent face data the encoder has never seen. Because both the dataset and encoder differ from our text and CLIP experiments, the comparison does not show which change causes the lower breakpoints. Together with the MS-MARCO and CLIP/MS COCO results, this shows the same attack working on text, multimodal, and face embedding pipelines.

n	k	Advantage	Accept (M/N)	AUC
100	2	0.600 ± 0.033	0.802/0.202	0.887
1000	3	0.718 ± 0.035	0.912/0.194	0.924

Table 6: Matched RAP attack on FaceNet embeddings of LFW face images at $\beta = 0.5$. Each row reports the first tested k at which mean advantage reaches 0.5. "M/N" gives member/nonmember acceptance. Error bars are standard errors across five seeds.

At the breakpoints reported in Section 6.5 the attack accepts between 0.81 and 0.92 of members and between 0.13 and 0.33 of nonmembers. The member acceptance rate barely moves across the six sizes, so what changes with the database is how often the attack accepts a nonmember. One known plaintext below the breakpoint at $n = 100$, the nonmember acceptance rate is 0.48, so the attack there accepts nearly half the nonmembers it is shown.

Raising the noise to $\beta = 1$ raises the breakpoint to $k = 6$ at $n = 500$ and $k = 7$ at $n = 2{,}000$. At $n = 2{,}000$ the preceding k has advantage 0.492 ± 0.027, which we cannot distinguish from 0.5, so the breakpoint there may in fact be 6.

The game samples the known plaintexts together with the database, so the adversary does not choose them. Letting it choose them by any of three selection rules does not lower the breakpoint at $n = 100$ or $n = 500$, and sorting the challenges by how isolated or how crowded each record is shows no trend in advantage either (Appendix F.2).

On CLIP/MS COCO, the attack reaches advantage 0.564 ± 0.025 at the breakpoint $k = 3$ and 0.214 ± 0.034 at $k = 2$, accepting 0.87 of members and 0.31 of nonmembers, with AUC 0.861.

Relation to orthogonal-transform recovery. The adversary holds the kind of input Giannella et al. [GLK13] use to recover an orthogonal transform, but recovering M is neither necessary nor available to it. The k pairs do not determine M when k is much smaller than the dimension, and the noise makes any fit approximate. Deciding membership may also be easier than recovering M.

B.3 Interpreting the Matched Bound

This subsection interprets the bound of Theorem 6.4. At $k = 0$, the sketch consists only of the noisy norm. Its distribution does not depend on the fixed unit vector q, so $\nu_q = \mu$ and $\delta_{d,0,N,\beta} = 0$, recovering Theorem 6.3.

The constants in the theorem are absolute, and we may increase c without weakening the bound; in particular, take $c > 2$, so $c/\beta > 1$ throughout the stated range. For any fixed target advantage α, the right-hand side is at most α whenever

$$k \le \frac{\log(n-k) - 2\log(C/\alpha)}{\log(c/\beta)},$$

which is the sense in which reaching a fixed constant advantage requires $k = \Omega(\log n)$. We do not attach numerical values to the proof constants, because the Beta and Gamma estimates of Lemma A.6 intentionally absorb absolute factors into c and C. Appendix A.5 instead computes the quantity the theorem bounds directly, which gives numbers without needing c or C.

Scope and limitations. Real embeddings are not uniform, so the theorem does not apply to the databases our attacks run on. It still limits what a distribution-independent attack can do. Any attack keeping constant advantage for every input distribution must also succeed on

the uniform sphere, and there the bound goes to zero as the database grows at fixed $\beta \in (0, 2]$ and k. Breaking RAP on realistic embeddings therefore requires using the non-uniformity of the embedding distribution, which the attempts of Section 6.4 try to do. The uniform sphere is also a hard nearest-neighbor workload, reaching Index Recall@10, the recall of the index against exact search on the same vectors, of only 0.291 over 10^5 plaintext vectors (Appendix F.3). The bound leaves little margin over our own attacks either. At $n = 10^4$ it says nothing from $k = 5$ on, which is where our matched attack reaches advantage 0.5 on MS-MARCO. These are different distributions, so what the theorem establishes is that needing $k = \Omega(\log n)$ is a property of the scheme and not an artifact of our search budget.

The bound covers the comparison-only leakage profile at $\beta > 0$ (Section 8), which is simulatable from the noisy Gram matrix. It does not cover the noiseless $\beta = 0$ settings for the comparison-only leakage profile and encrypted HNSW index. Without noise a single known plaintext already breaks RAP, as Section 6.6 notes, so the bound becomes vacuous as β tends to 0. Projected RAP is excluded for a different reason: it adds noise after projection, so its leakage is not a post-processing of the noisy Gram matrix in $\mathbb{R}^d$.

B.4 Unmatched Attack Measurements

The two error probabilities. Write $\mathcal{A}_{B,t}$ for the unmatched attack with beam width B and threshold t. Its advantage is the difference between the probabilities that $\widehat{s}_B(p) \leq t$ when $b = 1$ and when $b = 0$. By Claim B.1, when $b = 1$ and the beam retains the assignment mapping each known plaintext to its actual position, every inner-product error for that choice is at most β and hence $\widehat{s}_B(p) \leq \beta$. For $t \geq \beta$ the member acceptance probability is then at least the probability that the beam retained the actual assignment. A false positive instead happens when the attack finds some assignment together with an unassigned position that fits all the required inner products by accident, and how often that happens depends on the input distribution as well as on n, k and β.

The attack does not recover the matching. When $b = 1$ the challenge is itself a database vector whose position the adversary is not told, so an adversary that locates that position decides b, and membership inference is in this sense no harder than recovering the matching. The lowest-error matching is the actual one in only 22% of the held-out runs at $(n, k) = (50, 4)$, where the advantage is 0.48, and in 45% at $(n, k) = (100, 5)$, where the advantage is 0.60. The membership test is stronger than matching recovery because it retains several plausible matchings and tests the challenge jointly against all of them.

Scaling of the attack breakpoint. A wrong assignment that the attack nevertheless accepts must reproduce all $\binom{k}{2}$ pairwise inner products of the known plaintexts within the attack's tolerance, and there are fewer than n^k assignments to rule out, so the requirement should grow like $\log n$. On inputs uniform on the sphere at $\beta = 0.5$ the measured breakpoint is $k = 5, 6, 7, 9$ for $n = 50, 100, 200, 400$, which is consistent with that, although at the largest sizes advantage tracks how often the beam retains the true matching as much as it tracks the scheme.

Search budget. The attack searches over matchings, so its success depends on the effort spent finding one with lower error. Widening the beam from 5000 to 50,000 at $n = 1{,}000$ lowers the breakpoint to eight, so our breakpoints are upper bounds and a better-resourced adversary does better than we report. A test that does no searching at all is stopped by the noise, so shuffling alone does not stop such an attack whereas noise and shuffling together do.

A test that does no searching. Comparing a statistic of the whole hashed database against the same statistic with the challenge inserted avoids the search entirely, and the noise stops it. On MS-MARCO it reaches advantage 1.000 at both $n = 10^4$ and $n = 10^5$ when $\beta = 0$, but only 0.020 and 0.007 at $\beta = 0.5$, with AUC 0.532 and 0.506.

B.5 Unmatched Attack: Setup and Sweeps

Experiments. We evaluate the attack on the same 384-dimensional normalized MS-MARCO embeddings used above and on vectors sampled uniformly from the unit sphere. For each game, we sample a database of size n, give the attack k of its plaintexts, hash the database with fresh uniform-ball noise of radius $\beta/4$, and shuffle the positions. We compute one score for a member challenge and one for an independent nonmember challenge. The attack receives the known plaintexts, the challenge, and the noisy Gram matrix of the shuffled database. It is not given any of the hidden links between known plaintexts and positions.

We generate the leakage with $M = I$, which is without loss of generality because the noisy Gram matrix has the same distribution for every M. Thresholds are chosen on separate calibration runs and evaluated on disjoint held-out runs, each giving one member and one nonmember score, and we combine their signed acceptance-rate gaps as described in Section 4.3. The run counts, seeds, and beam width are in Appendix F.1.

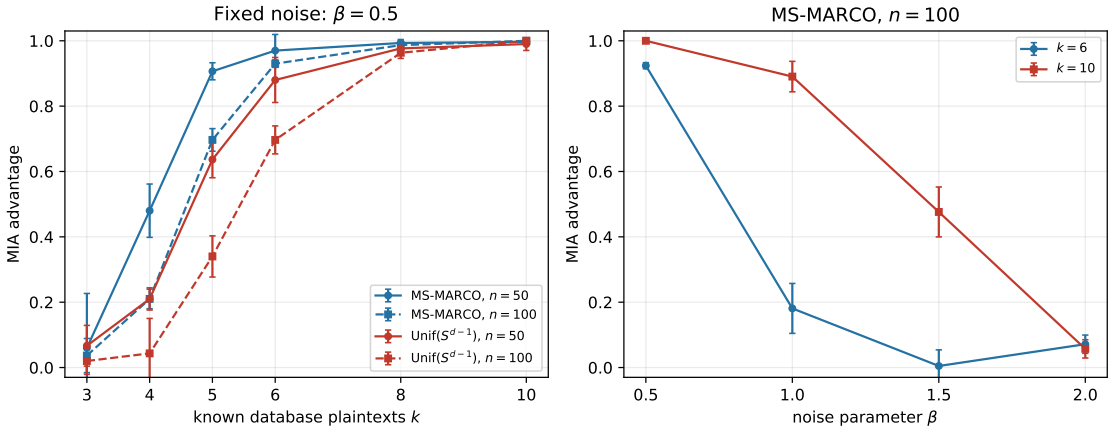

Figure 6: Unmatched known-plaintext attack. The left panel varies the number of known plaintexts with $\beta = 0.5$. The right panel varies the noise parameter on MS-MARCO with $n = 100$. Error bars here show 1.96 standard errors across the per-seed signed gaps, a 95% interval; every other figure in this paper shows one standard error.

In the left panel of Figure 6, the advantage rises steeply with k. On MS-MARCO with $n = 50$, the advantage is 0.48 for $k = 4$, 0.91 for $k = 5$, and 0.97 for $k = 6$. With $n = 100$, the corresponding advantages are 0.21, 0.70, and 0.93; we measured the $n = 100$ values quoted in Section 6.6 with five seeds and a fresh key drawn for every run, and they appear in Appendix C.5. The experiment on vectors uniform on the sphere shows the same pattern, although it requires slightly more known plaintexts. Its advantage is 0.64 for $(n, k) = (50, 5)$, accepting 0.80 of members and 0.16 of nonmembers, and 0.70 for $(n, k) = (100, 6)$, accepting 0.82 and 0.12; it exceeds 0.96 for both database sizes when $k = 8$. We conclude that our attack is not specific to the distribution of MS-MARCO embeddings.

The transition is almost entirely in the nonmember acceptance rate. At $n = 100$ it falls from

0.41 at $k = 4$ to 0.28 at $k = 5$ and 0.03 at $k = 6$, while the member acceptance rate rises from 0.62 to 0.99. At $k = 4$ the attack accepts more than half the nonmembers it is shown, so its advantage of 0.21 reflects an attack that accepts nearly everything rather than one that discriminates weakly.

The right panel isolates the effect of adding noise. We measured it separately from the left panel, with the run counts in Appendix F.1, so its $\beta = 0.5$ point is 0.92 rather than the left panel's 0.93. With $n = 100$ and $k = 6$, increasing β from 0.5 to 1 reduces advantage from 0.92 to 0.18, and at $\beta = 1.5$ and 2 the measured advantage falls below 0.08. These two measurements are within error of each other, so we do not infer an ordering. However, with ten known plaintexts it remains 0.89 at $\beta = 1$ and 0.48 at $\beta = 1.5$. Noise therefore increases the number of known plaintexts the attack needs to reach high advantage, but this comes at the cost of greater allowed distance-comparison error.

B.6 Worst-Case Correctness on the Sphere

FGHO [FGHO22] use the worst-case β-DCPH guarantee. It applies to every triple whose gap exceeds β, for every key and choice of hash coins, where the *gap* of a triple (x, y, z) is

$$\mathsf{gap}(x, y, z) = \big|\mathsf{dist}(x, y) - \mathsf{dist}(x, z)\big|.$$

This distribution-free guarantee lets us compare RAP with SAP on the same terms.

We measure how often this guarantee applies at $d = 384$ using five seeds and exactly 200,000 sampled distinct ordered triples per seed. For 12,000 fresh uniform-sphere vectors per seed and the fixed 4,100-vector SciFact pool, the mean per-seed median gaps are respectively 0.03450 ± 0.00004 and 0.08154 ± 0.00010. At $\beta = 0.5$, the covered fractions are 0 and $0.113\% \pm 0.005\%$; at $\beta = 0.05$, they are $32.801\% \pm 0.038\%$ and $67.317\% \pm 0.030\%$. These are standard errors across seeds and measure worst-case-guarantee coverage, not retrieval quality.

Retrieval correctness is measured separately. On MS-MARCO at $\beta = 0.5$, Neighbor Recall@10 is 0.929, and at $\beta = 1$ it is 0.868, both at $n = 10{,}000$ over three seeds. We use Neighbor Recall@10 and task-level retrieval metrics for empirical correctness, and leave open a DCPH correctness notion tailored to this setting.

C Secret Projection

This appendix collects the material on projected RAP from Section 7. It gives the leakage-equivalence proof, the second-moment calculation the proof uses, the task-aware variant, and the attack and utility measurements.

C.1 Effect on the Attacks

Projection makes the Gram-pattern constraints used by the beam search less exact, so more assignments stay plausible and the attacks need more known plaintexts. Proposition 7.3 treats one pair of vectors in isolation, whereas every pair in the hashed database shares the same secret subspace and so has correlated projection error. We therefore tried a score that accounts for this. It ranks the first 300 retained assignments using the exact covariance of the projection error across pairs, computed as in Appendix C.3, together with a diagonal approximation to the hashing noise. We ran it under four seeds and report the absolute value of the seed-averaged signed gap between the member and nonmember acceptance rates, which is the advantage of Definition 4.5. It did not improve on the prespecified attacks. At $m = 128$ it reached 0.018 at $k = 20$ and 0.010 at $k = 24$,

against 0.070 for the conditional residual attack. We do not treat this exploratory comparison as a new calibrated adversary.

C.2 Projected RAP Leakage Equivalence

Proof of Lemma 7.2. Write the projected-RAP hash tuple as the $m \times n$ matrix

$$H = (h_1, \ldots, h_n), \qquad h_i = \sqrt{d/m}\, Q^\top x_i + \eta_i,$$

where the columns of $Q \in \mathbb{R}^{d\times m}$ are a uniformly random orthonormal m-frame and the η_i are independent and uniform in the radius-$\beta/4$ ball of $\mathbb{R}^m$. The simulator from hash values to projected Gram leakage returns $G = H^\top H$.

For the reverse direction, first observe that the real hash tuple has no preferred output coordinate system. If $R_0 \in O(m)$ is fixed, then

$$R_0 H = \sqrt{d/m}\, (Q R_0^\top)^\top X + R_0 \Xi,$$

where $X = (x_1, \ldots, x_n)$ and $\Xi = (\eta_1, \ldots, \eta_n)$. The matrix $QR_0^\top$ has the same distribution as Q, and rotating uniform ball noise leaves its distribution unchanged. Hence

$$R_0 H \equiv H \qquad \text{for every fixed } R_0 \in O(m).$$

The same is true for an independent uniform R_0.

We also use the elementary fact that two configurations with the same Gram matrix differ by one common orthogonal transformation. Indeed, if $Y, Y' \in \mathbb{R}^{m\times n}$ satisfy $Y^\top Y = (Y')^\top Y'$, the map sending each column of Y to the corresponding column of Y' is a well-defined isometry on their column spans and extends to some $S \in O(m)$.

On input a positive semidefinite $G \in \mathbb{R}^{n\times n}$ of rank at most m, the simulator computes any factor $Y(G) \in \mathbb{R}^{m\times n}$ with $Y(G)^\top Y(G) = G$, samples a uniform $R \in O(m)$, and returns $RY(G)$. For example, it may use the spectral factorization from Lemma 6.2.

It remains to check the distribution. Let h be any realization of the real hash tuple with Gram matrix G. By the preceding fact there exists an orthogonal S_h such that $h = S_h Y(G)$. For any measurable set A,

$$\Pr[Rh \in A \mid H = h] = \Pr[RS_h Y(G) \in A] = \Pr[RY(G) \in A],$$

where the last equality is right invariance of the uniform distribution on $O(m)$. The right-hand side depends on h only through G. Averaging over all realizations h with that Gram matrix therefore shows that, conditional on G, RH has exactly the simulator's distribution. But $RH \equiv H$ and the common rotation does not change the Gram matrix, so the same is true of the real hash tuple H. Thus the simulator is exact. □

C.3 Secret-Projection Calculation

The proof of Proposition 7.3 uses the second moments of a uniformly random rank-m projection matrix P, by which we mean the expectations $\mathbb{E}[P_{ij}P_{k\ell}]$ of products of two entries of P. We derive them here using only three facts about P, each of which can be checked directly. We use no result from representation theory.

The three facts are these. First, P is symmetric, so $P_{ij} = P_{ji}$. Second, $P = QQ^\top$ where the columns of Q span a uniformly random m-dimensional subspace, so for every fixed orthogonal matrix R the matrix $R^\top PR$ has the same distribution as P. Third, $\operatorname{tr}(P) = m$ and $P^2 = P$ hold for every value of P, not just on average.

Step 1: which moments vanish. Fix a coordinate s and let D be the diagonal matrix with -1 in position s and $+1$ elsewhere. Then D is orthogonal, so DPD has the same distribution as P. Entrywise, $(DPD)_{ij} = d_i d_j P_{ij}$ where $d_s = -1$ and every other $d_i = +1$. Therefore

$$\mathbb{E}[P_{ij}P_{k\ell}] = d_i d_j d_k d_\ell \, \mathbb{E}[P_{ij}P_{k\ell}].$$

If the index s occurs an odd number of times in the list i, j, k, ℓ, then $d_i d_j d_k d_\ell = -1$ and $\mathbb{E}[P_{ij}P_{k\ell}]$ equals its own negative, so it is zero. A nonzero moment therefore requires every coordinate to occur an even number of times in that list. The list has length four, so those multiplicities are even and add to four. Either one coordinate occurs four times, or two distinct coordinates occur twice each. There is no other possibility.

Step 2: how many distinct values remain. Let R be a permutation matrix. It is orthogonal, so $R^\top PR$ has the same distribution as P, and $R^\top PR$ is P with its coordinates renamed. The value of a moment therefore depends on the arrangement of the indices and not on the particular coordinates involved. Using $P_{ij} = P_{ji}$ to identify the two paired arrangements, three numbers remain,

$$p = \mathbb{E}[P_{11}^2], \qquad q = \mathbb{E}[P_{11}P_{22}], \qquad r = \mathbb{E}[P_{12}^2].$$

Here p is the all-equal case, q is the case $i = j$ and $k = \ell$ with $i \neq k$, and r is the case $i = k$ and $j = \ell$ with $i \neq j$.

Step 3: the three numbers satisfy $p = q + 2r$. This step is what lets a single formula cover the all-equal case as well as the paired cases. Let $u = (e_1 + e_2)/\sqrt{2}$ and let R be the rotation with $Re_1 = u$ that acts only on the first two coordinates. Since $R^\top PR$ has the same distribution as P, the number $(R^\top PR)_{11}$ has the same distribution as P_{11}. Also

$$(R^\top PR)_{11} = (Re_1)^\top P(Re_1) = u^\top Pu = \tfrac{1}{2}\big(P_{11} + P_{22} + 2P_{12}\big),$$

using $P_{12} = P_{21}$. Squaring both sides and taking expectations,

$$p = \mathbb{E}[P_{11}^2] = \tfrac{1}{4}\,\mathbb{E}\big[(P_{11} + P_{22} + 2P_{12})^2\big].$$

Expand the square. The terms $\mathbb{E}[P_{11}P_{12}]$ and $\mathbb{E}[P_{22}P_{12}]$ are zero by Step 1, because in each the second coordinate index occurs an odd number of times. What remains is

$$p = \tfrac{1}{4}\big(p + p + 2q + 4r\big),$$

which rearranges to $p = q + 2r$.

Step 4: one formula for every index pattern. The three cases of Step 1 can be written as one formula,

$$\begin{aligned}\mathbb{E}[P_{ij}P_{k\ell}] = {} & q\,\mathbf{1}[i = j]\mathbf{1}[k = \ell] \\ & + r\,\mathbf{1}[i = k]\mathbf{1}[j = \ell] \\ & + r\,\mathbf{1}[i = \ell]\mathbf{1}[j = k].\end{aligned}$$

Check it against each case of Step 1. When the indices do not pair up, every indicator is zero and both sides vanish. In the case counted by q the first term contributes q and the others vanish. In the case counted by r one of the last two terms contributes r and the others vanish. When all four indices are equal all three indicators equal one, so the right side is $q + 2r$, which equals p by Step 3. The formula therefore holds for every choice of i, j, k, ℓ. Step 6 uses this identity.

Step 5: fixing the two constants. The constants follow from the two identities that hold for every value of P. First,

$$m^2 = (\operatorname{tr} P)^2 = \sum_{i,k} P_{ii} P_{kk}.$$

Taking expectations and substituting the formula above gives

$$d^2 q + 2dr = m^2.$$

Second,

$$m = \operatorname{tr}(P^2) = \sum_{i,j} P_{ij}^2,$$

which gives

$$dq + d(d+1)r = m.$$

Solving these two linear equations gives

$$r = \frac{m(d-m)}{d(d-1)(d+2)}, \qquad q = \frac{m\big(m(d+1)-2\big)}{d(d-1)(d+2)}.$$

The proof of Proposition 7.3 only needs r explicitly. As a quick check, when $m = d$ we have $P = I$ deterministically, and these formulas give $r = 0$ and $q = 1$, as they should.

Step 6: the variance. For fixed $x, y \in \mathbb{R}^d$, write $c = \langle x, y \rangle$. Expanding $x^\top P y$ twice and applying the moment identity gives

$$\mathbb{E}[(x^\top P y)^2] = qc^2 + r\big(\|x\|^2 \|y\|^2 + c^2\big).$$

Since $\mathbb{E}[P] = (m/d) I$,

$$\mathbb{E}[x^\top P y] = \frac{m}{d} c.$$

Subtracting the square of this mean and multiplying by $(d/m)^2$ gives

$$\operatorname{Var}\left(\frac{d}{m} x^\top P y\right) = \frac{(d-m)\big(d\|x\|^2 \|y\|^2 + (d-2)c^2\big)}{m(d-1)(d+2)}.$$

Thus the projection-variance formula follows from sign changes, coordinate renamings, one rotation, and two identities that hold for every value of P. No concentration theorem is used.

This proves Proposition 7.3. Taking $x = y = u$ gives

$$\frac{\operatorname{Var}\Big(\frac{d}{m}\big\|Q^\top u\big\|^2\Big)}{\|u\|^4} = \frac{2(d-m)}{m(d+2)},$$

which is the distance-distortion calculation used in Section 7.

Relation to Johnson–Lindenstrauss. The projection used here is the one in the Johnson–Lindenstrauss lemma [JL84], a uniformly random m-dimensional subspace with the scaling $\sqrt{d/m}$. The proposition is a different statement about that projection. Johnson–Lindenstrauss bounds the probability that some pair among n points has its distance changed by more than a factor $1 \pm \varepsilon$. The proposition gives the exact mean and the exact variance for one pair of vectors. A bound of the Johnson–Lindenstrauss form does not give the exact variance, and the variance on its own gives only what Chebyshev's inequality provides. We need the exact variance because Section 7.3 uses it as the noise scale in the attack's scoring. As a check on the special case $x = y$ with $\|x\| = 1$ and $m < d$, the quantity $\big\|Q^\top x\big\|^2$ has the Beta$(m/2, (d-m)/2)$ distribution, whose variance is $2m(d-m)/(d^2(d+2))$, and multiplying by $(d/m)^2$ gives the formula above.

C.4 Task-Aware Projection

This appendix gives the construction and results summarized in Section 7.

A random projection treats every direction alike, so we also tried choosing the directions from public data. A public map $W \in \mathbb{R}^{d \times r}$ learned from a public split of the corpus is handed to the adversary, adding no secret randomness of its own, and the secret projection is applied inside that representation. It recovers retrieval quality, raising nDCG@10 on SciFact from 0.605 to 0.612 at matched projection error, but it does not improve security and instead reduces it: at $k = 20$ the attack reaches advantage 0.613 against the task-aware map and 0.392 against the random projection to 192 dimensions. The public map therefore improves retrieval quality but not privacy.

Calibration and reporting. We report the larger measured gap of the two attacks. Under each of four seeds, 50 calibration games set the thresholds and 100 fresh paired member/nonmember games measure the signed difference between the two acceptance rates. Every game draws its own secret projection, database, and noise, so the thresholds are never calibrated against the key they are later applied to and the measured gap estimates MIA advantage as Definition 4.5 defines it. We average the four signed differences before taking the absolute value; the standard error across them is at most 0.06 in Table 7, although four seeds provide only a rough estimate. We therefore do not treat either a one-step difference in the first k reaching 0.5 or a difference below about 0.1 as meaningful; taking the larger measured gap also favors the attack.

Construction	Dimensions	Projection SD (c=0)	nDCG@10	MRR@10	Recall@100
Unprotected all-MiniLM-L6-v2	384	—	0.645	0.605	0.925
Projected RAP	$384 \to 128$	0.072	0.571	0.532	0.884
Task-aware projected RAP	$384 \to 192 \to 96$	0.072	0.580	0.545	0.865
Projected RAP	$384 \to 192$	0.051	0.605	0.565	0.902
Task-aware projected RAP	$384 \to 192 \to 128$	0.051	0.612	0.578	0.890
Task-aware, no secret reduction	$384 \to 192 \to 192$	0.000	0.638	0.604	0.904

Construction	Dimensions	Advantage at $k = 20$	First k with advantage ≥ 0.5
Projected RAP	$384 \to 128$	0.055 ± 0.045	> 24
Task-aware projected RAP	$384 \to 192 \to 96$	0.068 ± 0.011	> 24
Projected RAP	$384 \to 192$	0.392 ± 0.030	24
Task-aware projected RAP	$384 \to 192 \to 128$	0.613 ± 0.027	20
Task-aware, no secret reduction	$384 \to 192 \to 192$	1.000 ± 0.000	6

Table 7: Utility and measured attack advantage for task-aware projected RAP on SciFact. "Projection SD (c=0)" is the standard deviation given by Proposition 7.3 for orthogonal unit vectors; it is the projection-induced spread only and excludes noise. "First k with advantage ≥ 0.5" is the smallest tested number of known database vectors for which either attack reaches advantage 0.5; > 24 means that neither attack does so through $k = 24$. Each seed uses 100 paired held-out games with a fresh secret projection in every game; we average the four signed acceptance-rate differences before taking the absolute value, and report the standard error across those four signed differences. Advantage differences below about 0.1 and one-step differences in the first breakpoint k are within this variation.

Repeating this comparison with `all-mpnet-base-v2`, which has input dimension 768, in place of `all-MiniLM-L6-v2` gives the same ordering, with the task-aware map reaching a breakpoint at $k = 16$ and the random projection not reaching one through $k = 24$. We give the indexed utility measurements in Appendix F.3.

C.5 Secret Projection: Attack and Utility Measurements

Attack and utility experiments. We test projected RAP on normalized MS-MARCO embeddings of dimension 384, with $\beta = 0.5$. The membership experiment uses databases of size $n = 100$ and the unmatched setting, which does not reveal which position belongs to each known plaintext. We measure the effect of projection at this one database size, so what follows says how much projection raises the breakpoint at $n = 100$ and not how that amount changes as the database grows.

For each output dimension m and number k of known database plaintexts, we run the two attacks of Section 7.3: the joint Gram-pattern attack and the conditional residual attack. Separate calibration runs choose which attack and threshold to use, so this choice does not depend on the test results. Every game samples a fresh secret projection, database, noise, and shuffling permutation. We average the signed acceptance-rate gap across seeds before taking its absolute value to obtain

MIA advantage. The run counts, seeds, and beam width are in Appendix F.1.

We measure retrieval quality separately on databases of size 10,000, where queries are searched using the plaintext and projected distances, and Neighbor Recall@10 is the average fraction of the plaintext top-10 neighbors that remain in the projected top 10. This metric uses no relevance labels, so it shows how closely projected RAP reproduces ordinary nearest-neighbor search rather than the quality of the underlying embedding model.

Retrieval quality at deployment scale. Projection changes the distances used for retrieval. For every difference vector $u = x - y$,

$$\mathbb{E}\left[\frac{d}{m}\left\|Q^\top u\right\|^2\right] = \|u\|^2 .$$

When $u \neq 0$, the relative variance is

$$\frac{\mathrm{Var}\left(\frac{d}{m}\left\|Q^\top u\right\|^2\right)}{\|u\|^4} = \frac{2(d-m)}{m(d+2)}.$$

Neighbor Recall@10 asks what changes relative to plaintext search and counts a swapped-in document as a loss even when it is equally relevant. We therefore also measure MRR@10 against the relevance judgments of the MS-MARCO development split, over 6,980 queries, on databases of 10^4, 10^5, and 10^6 vectors at $\beta = 0.5$. Each database contains every judged passage for those queries, and each smaller database is a subset of the larger ones, so these numbers are not the official full-corpus MS-MARCO MRR@10 and are comparable only across our own protected and plaintext runs. We report retention, the protected MRR@10 divided by the plaintext MRR@10 on the same database, since plaintext retrieval itself gets harder as the corpus grows. At $m = 256$ retention is 0.994, 0.988, and 0.983 across the three sizes, at $m = 192$ it is 0.988, 0.978, and 0.969, and at $m = 128$ it is 0.973, 0.955, and 0.934, so the cost grows with the database but stays under two percent at the $m = 256$ setting. On these nested corpora, Neighbor Recall@10 at $m = 256$ rises from 0.742 at 10^4 to 0.789 at 10^6; the 0.767 of Table 1 is measured on the random 10,000-vector databases of Appendix F.1, so the two are not the same experiment.

Random-projection results. Table 1 contains three quantities per output dimension m. The projection SD is the standard deviation of Proposition 7.3 for two orthogonal unit vectors, so it measures the projection error alone and excludes the added RAP noise. The attack breakpoint is the first tested k at which held-out fresh-key MIA advantage reaches 0.5, and > 99 means it is not reached through $k = 99$. The unprotected row compares exact plaintext search with itself, so it has no breakpoint.

As m decreases in Table 1, the projection-induced error predicted by Proposition 7.3 grows, the tested attacks require more known plaintexts, and retrieval quality also decreases. Thus m must be selected to balance utility and security accordingly. Because every row draws a fresh key for every run, the ordinary-RAP row at $m = 384$ is that scheme measured the same way as the projected ones, so it belongs to the same trend as its $m = d$ endpoint. At $m = 128$, the attack does not reach a breakpoint through $k = 99$, but its advantage there is still 0.389 and its AUC is 0.672. Its member acceptance rate is 0.42 against a nonmember acceptance rate of 0.03. The held-out set has equal numbers of members and nonmembers, so almost everything the attack accepts is a member, and it simply accepts few of them. If members were rare among the candidates, most accepted items would be nonmembers. This endpoint is therefore not evidence of security; it records only the largest value of k tested.

Projection at a larger database size. Table 1 is at $n = 100$. Repeating the sweep at $n = 1{,}000$ raises every breakpoint. Without projection the unmatched attack reaches one at $k = 12$ rather than $k = 5$. At $m = 256$ it reaches one at $k = 128$ rather than $k = 18$, accepting 0.64 of members and 0.00 of nonmembers there. At $m = 192$ and $m = 128$ it reaches no breakpoint through the $k = 256$ we test, and the two differ in the same way as at $n = 100$. At $m = 192$ the advantage is 0.474 ± 0.035 with AUC 0.624, so the score orders members above nonmembers and the attack is close to reaching a breakpoint. At $m = 128$ the advantage is 0.034 ± 0.031 with AUC 0.487, no better than chance. Under our fixed beam-search budget, the gain from projection in breakpoint therefore grows with the database. At $m = 256$ it multiplies what the attack needs by about three and a half at $n = 100$, and by about ten at $n = 1{,}000$.

D Additional Comparison-Only Experiments

D.1 Scope and Common Setup of the Experiments

These experiments use $n = 100$; they compare leakage profiles at one database size and do not establish how the breakpoints grow with it.

The subsection below gives the full-dimensional sweeps. The projected results at each output dimension, and the settings that reach no breakpoint, are in Section 8.

D.2 Comparison-Only Results at Full Dimension

Our experiments use database size $n = 100$. Separate calibration runs select the prefix width and threshold, and disjoint paired held-out runs estimate advantage. Rank predictions use a public reference sample disjoint from every challenge database. The run counts, seeds, beam width, and reference-sample size are in Appendix F.1.

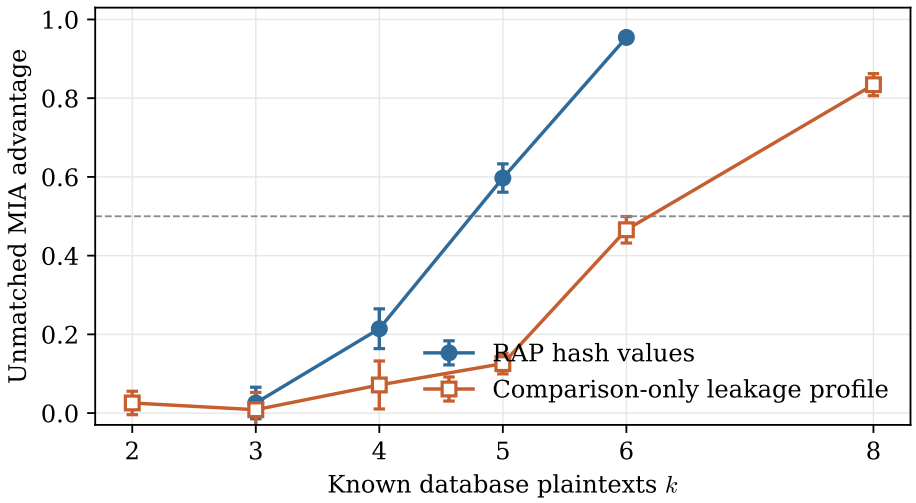

Figure 7: Full-dimensional RAP and the comparison-only leakage profile on MS-MARCO with $\beta = 0.5$ and $n = 100$, using the unmatched attack. Error bars give one standard error across five seeds.

We compare RAP with the comparison-only leakage profile in Figure 7 at $\beta = 0.5$ and $n = 100$. In every comparison-only trial, the leakage profile is computed from perturbed vectors generated by

the RAP mechanism and therefore has exactly their retrieval order. The two attacks use independent trials but the same data distribution, noise mechanism, and parameters. Their top-10 Neighbor Recall is 0.929 at $n = 10{,}000$. In this controlled experiment, removing coordinates and numerical distances raises the breakpoint of the tested unmatched attack from $k = 5$ to $k = 8$, without changing the retrieval order. At $k = 8$ the attack accepts 0.89 of members against 0.05 of nonmembers.

	Unmatched		Matched	
Noise β	Advantage	Nonmember	Advantage	Nonmember
0	0.99	0.01	0.99	0.01
0.5	0.91	0.02	0.99	0.01
1	0.38	0.32	0.92	0.05
1.5	< 0.01	0.70	0.74	0.15
2	0.03	0.42	0.52	0.21

Table 8: Effect of shuffling on the comparison-only attack at database size $n = 100$, with the number of known database vectors held fixed at $k = 10$. Entries are MIA advantage and the nonmember acceptance rate at the same threshold, not breakpoints. "Matched" gives the adversary the position of each known vector in the leakage profile, while "unmatched" does not reveal these positions. Values use 210 paired held-out runs across three seeds, with standard error at most 0.05.

Table 8 shows the effect of hiding the plaintext-to-position correspondence alone, holding the number of known vectors fixed at $k = 10$ and varying the noise. With the correspondence hidden, the unmatched attack degrades sharply as noise grows, while the matched attack retains high advantage throughout. This is the effect that record–encoding unlinkability relies on.

E Encrypted HNSW: Proof and Experimental Details

E.1 HNSW Reduction

Lemma E.1 (Idealized HNSW state from Gram leakage)**.** *For the exact-arithmetic distance-matrix constructor* $\mathsf{HNSWBuild}_{\theta_{\mathrm{H}}}$ *of Definition 9.1,* $\mathcal{L}_{\mathrm{HNSW},\theta_{\mathrm{H}}}$ *is simulatable from* $\mathcal{L}_{\mathrm{Gram},0}$*.*

Proof. Let $G = \mathcal{L}_{\mathrm{Gram},0}(S)$. Compute the squared-distance matrix Δ by

$$\Delta_{ij} = G_{ii} + G_{jj} - 2G_{ij}.$$

This is exactly $\Delta(S)$. Sample $\tau \leftarrow\!\!\$\ \mathsf{Perm}(n)$ and $\omega \leftarrow\!\!\$\ \mathcal{R}_{\theta_{\mathrm{H}},n}$ as in Definition 9.1, which the simulator can do because $\mathcal{R}_{\theta_{\mathrm{H}},n}$ depends only on public parameters and n, form

$$\Delta^{\tau}_{ab} = \Delta_{\tau^{-1}(a),\tau^{-1}(b)},$$

and run

$$S_{\tau} \leftarrow \mathsf{HNSWBuild}_{\theta_{\mathrm{H}}}(\Delta^{\tau}; \omega).$$

Relabel build vertex $\tau(i)$ by database position i and return $\big(\mathrm{lev}_{\tau}, (E_{\tau,r})_{r=0}^{L_{\tau}}, e_{\tau}\big)$. Every random choice and every input to the constructor now has exactly the distribution specified by Definition 9.1, so the returned distributions are identical. □

Security inherited by post-processing. Lemma E.1 is useful beyond comparing leakage profiles. Write $H(S) = (Mx_1 + \eta_1, \ldots, Mx_n + \eta_n)$ for the RAP hash values with noise parameter β, and define the profile that builds the idealized index on them,

$$\mathcal{L}_{\mathrm{HoR},\theta_{\mathrm{H}},\beta}(S) = \mathcal{L}_{\mathrm{HNSW},\theta_{\mathrm{H}}}\big(H(S)\big),$$

where the hashing randomness and the construction randomness are independent.

We show this is a randomized post-processing of $\mathcal{L}_{\mathrm{Gram},\beta}$. Fix any value S' of $H(S)$. Applying Lemma E.1 to the database S' gives $\mathsf{Sim}_{\mathrm{HNSW}}(\Gamma(S')) \equiv \mathcal{L}_{\mathrm{HNSW},\theta_{\mathrm{H}}}(S')$. The proof of Lemma 6.2 shows that $\Gamma(H(S))$ has exactly the distribution of $\mathcal{L}_{\mathrm{Gram},\beta}(S)$; this is the statement $\Gamma(Mx_i + \eta_i) = \Gamma(x_i + \eta_i')$ established there, rather than the profile equivalence in the lemma statement. Averaging the previous display over the hashing randomness therefore gives $\mathsf{Sim}_{\mathrm{HNSW}}(\mathcal{L}_{\mathrm{Gram},\beta}(S)) \equiv \mathcal{L}_{\mathrm{HoR},\theta_{\mathrm{H}},\beta}(S)$.

Because this simulation holds for every ordered database, it also holds after conditioning on the outer shuffle in Definition 4.5. In the matched game the record-to-vertex pairs are auxiliary information and are passed through unchanged. Thus any membership-inference adversary against $\mathcal{L}_{\mathrm{HoR},\theta_{\mathrm{H}},\beta}$ can be converted, without loss in advantage, into an adversary against the noisy-Gram view in the same matched or unmatched known-plaintext setting by first running $\mathsf{Sim}_{\mathrm{HNSW}}$.

Consequently, every all-adversary security bound proved for RAP through the noisy-Gram profile also holds for this idealized HNSW-over-RAP snapshot. In particular, for uniform spherical data the perfect $k = 0$ result of Theorem 6.3 holds for every $\beta \geq 0$, and, again for uniform spherical data and $k < \min\{n, d\}$, the matched known-plaintext bound of Section 6.7 holds unchanged for $0 < \beta \leq 2$. For the FH-IPFE setting studied here, where the owner builds HNSW on the unperturbed database geometry rather than on noisy RAP hashes, Lemma E.1 gives post-processing only from $\mathcal{L}_{\mathrm{Gram},0}$. Thus the $\beta = 0$, $k = 0$ perfect-security consequence applies on uniform spherical data, but the positive-noise matched bound should not be attributed to that construction.

E.2 Implementation Details

Our experiments sample a fresh uniform insertion order for every Faiss HNSW index, matching the randomized insertion order in Definition 9.1. There is one representation difference. Faiss numbers vertices by insertion rank, whereas the formal leakage profile relabels the final graph back to protected-database positions and does not return the insertion permutation. The attacks reported in this paper do not use insertion rank: their scores are invariant under a common relabeling of the vertices. Thus relabeling each experimental graph into the formal representation would leave every reported score unchanged. We use the cleaner position-labeled profile in the formal analysis, while noting that a deployment that exposes insertion rank gives additional snapshot leakage not used by our attacks. The matched known-plaintext correspondence is supplied separately, as required by the matched game, rather than being part of the leakage profile.

Fitting the edge model. The implementation fits p_{edge} and p_0 separately at each HNSW layer and adds the layer scores. It bins the public inner products and clips the fitted values of $p_{\mathrm{edge}}(t)$ to $[10^{-3}, 1 - 10^{-3}]$ before taking logarithms.

Ordering the known plaintexts. Before starting the beam search, the attack orders the known vectors so that the most informative edge constraints are applied first. It begins with the pair whose predicted edge probability is farthest from p_0, then repeatedly adds the remaining vector whose predicted edge probabilities differ most in total from p_0 across the vectors already placed.

Placement recovery. We record the fraction of the true known-vector vertices contained in the highest-scoring assignment, whose expectation for a random set of k vertices is k/n.

Running times. Given the fitted curve, the matched score reads $2k$ edge bits for each of at most n candidate vertices and takes $O(nk)$ time. For the unmatched attack, the first placement step examines $n(n-1)$ ordered vertex pairs, the later steps examine at most Bn extensions and add $O(k)$ edge constraints per extension, and scoring the challenge under all retained placements takes $O(Bnk)$ time, so the complete search uses $O(n^2 + Bnk^2)$ arithmetic operations.

E.3 Evaluation Metric and Setup

Evaluation metric. For the calibrated matched sweep of Table 4 we report MIA advantage, as for every other attack in this paper; the small-index comparison and the unmatched results instead report empirical KS separation, defined below, which is not an advantage. The score these attacks produce scales with the number of known plaintexts, so a single threshold cannot serve the whole sweep, and we therefore choose the score direction and threshold independently at every setting of that sweep. Each setting uses three separate sets of games. The edge-probability curve is fitted on its own public databases, a second set of calibration games fixes the direction and threshold, and a third set of fresh held-out games estimates advantage, member and nonmember acceptance, and AUC. The decision rule is fixed before the held-out games are scored, so the reported advantage is that of a rule the adversary chose in advance.

We also record, on the same held-out scores, the largest member/nonmember acceptance-rate gap over all thresholds, which we call *empirical KS separation*. That quantity is at least the advantage of any fixed threshold, and even a score independent of membership shows some positive gap once every threshold is tried, so we use it only to check whether a score separates members from nonmembers at all, and compare it against a null reference obtained by swapping the member and nonmember labels at random within paired games. We do not report it as advantage anywhere.

Independent-sample reference. For two independent samples of size 5,000 from the same continuous distribution, the expected empirical KS separation is about $1.23/\sqrt{5{,}000} \approx 0.017$, since the two-sample Kolmogorov–Smirnov statistic at equal sample size T has mean $1.23/\sqrt{T}$ asymptotically. Our member and nonmember scores are not independent, since each pair comes from a shared database, graph, and set of known vectors, so 0.017 is only a reference value for this paired design. Where that approximation is not useful, we compare against the paired label-swap null of Appendix E.4.

Controls. The *matched known-plaintext control* runs the matched score of Section 9.1 with the true vertices corresponding to the known plaintexts supplied, as in the matched known-plaintext setting. The HNSW leakage profile itself is unchanged. The second control is the placement recovery defined above, whose random baseline is k/n. The search begins with a pair of known vectors, so it requires $k \geq 2$.

E.4 Experimental Evaluation of the HNSW Attacks

Here HNSW behaves differently from the leakage profiles studied above, which reveal more. For RAP, record–encoding unlinkability raises the tested breakpoint from three known plaintexts to five; for comparison-only leakage, from four to eight. For HNSW, by contrast, our unmatched edge-based attack remains at its null reference and its known-vertex recovery is essentially random even at

values of k where the corresponding matched attack is strong. This is an attack-specific negative result, not evidence that the HNSW snapshot is secure against all unmatched attacks.

We report two different kinds of numbers. The matched attack's breakpoints in Table 4 are MIA advantage, with the score direction and threshold fixed in advance on calibration games separate from the games scored. The small-index comparison in Table 9 and all the unmatched results instead use empirical KS separation, which picks the threshold after seeing the scores and so is not an advantage. We use it only where the finding is that a score does not separate members from nonmembers at all. A statistic chosen after the fact favors the attack, so a failure to separate under it is the stronger negative result.

Setup. We fit p_{edge} and p_0 on separate databases built from public vectors with the same public parameters, never on the challenge database. At $n = 200$ we use the HNSW parameters reported by Acharya et al. [AFLO25]: `M` $= 32$, `efConstruction` $= 30$, and `efSearch` $= 256$. We hold them fixed so that the only difference between the matched and unmatched runs is whether the plaintext-to-vertex correspondence is supplied. Table 9 contains the $k = 32$ results.

We repeat the comparison at $n = 5{,}000$ for the sphere and MS-MARCO and $n = 2{,}000$ for SciFact with `efConstruction` $= 100$, because the small index is dense and nearly single-layer whereas a larger index is sparse with several populated layers. The attack fits its edge model on three layers there. The run counts, seeds, and beam widths are in Appendix F.1.

Data	Unmatched attack	Matched attack	Known-vertex recovery
Sphere	0.018 ± 0.004	0.895 ± 0.002	0.161
MS-MARCO	0.014 ± 0.003	0.758 ± 0.001	0.162
SciFact	0.013 ± 0.001	0.121 ± 0.004	0.160

Table 9: HNSW attacks at $n = 200$ and $k = 32$. The two attack columns give empirical KS separation, not MIA advantage. Known-vertex recovery is the fraction of the k true vertices contained in the attack's highest-scoring size-k placement. A random placement has expected recovery $k/n = 0.160$. The independent-sample reference for empirical KS separation at 5,000 trials is 0.017; because the experimental design is paired, this is a reference rather than its exact null. Separation is the mean $\pm$ standard error over four seeds; recovery is their mean.

Matched known-plaintext attack results. The small-index results are in Table 9. At $n = 200$, the matched attack is well above the 0.017 reference on all three datasets, so the index state contains exploitable membership information when the known plaintext-to-vertex correspondence is supplied, although the strength varies considerably between SciFact and the sphere. This index is small and dense. With `M` $= 32$, only about $n/\texttt{M} = 6.25$ vertices are expected above the base layer, so almost all of the information used by this attack comes from one layer.

The larger indexes, in Figure 8, show the same ordering. We measured them separately from the calibrated sweep in Table 4, so the two sets of numbers are not comparable setting by setting. There the matched attack reaches empirical KS separation 0.449 with AUC 0.801 on MS-MARCO at $k = 1{,}024$ and crosses 0.5 on the sphere well below that, while on SciFact it reaches only 0.125 with AUC 0.579 at $k = 512$. Index Recall@10 is one at $n = 200$ and at least 0.997 on the larger indexes, so these index parameters do not materially degrade retrieval in our tests.

Scaling of the matched breakpoint. The matched attack needs far more known plaintexts against HNSW than against RAP. Against RAP the number grows about logarithmically in the

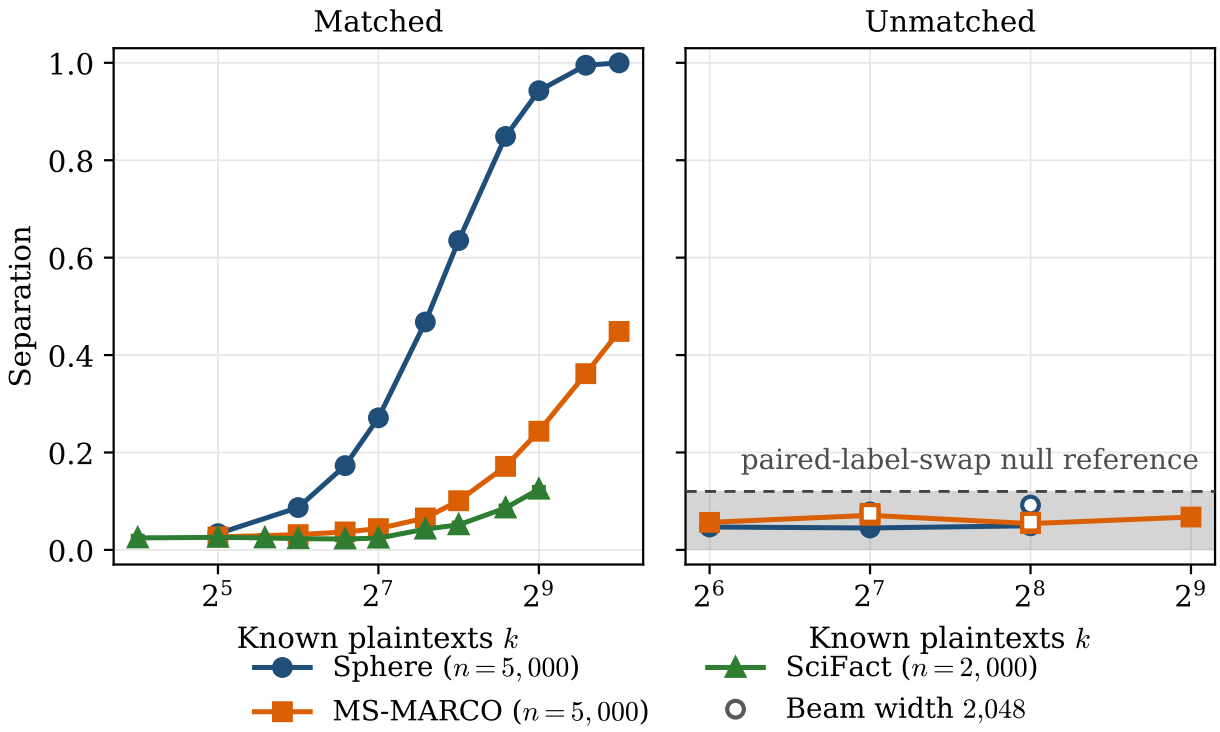

Figure 8: Empirical KS separation for the HNSW attacks with `M` = 32, `efConstruction` = 100, and `efSearch` = 256. Error bars give one standard error across four seeds. The shaded band in the right panel is the mean of the per-seed paired-label-swap null 95th percentiles, which is a reference scale rather than a calibrated percentile for the seed-averaged statistic. Open markers repeat two parameter settings at beam width 2,048. The sphere at $k = 512$ is omitted from the right panel because that setting completed only one of its four seeds.

database size; against HNSW on MS-MARCO it stays between a fifth and a third of the database over the sizes tested. Both sets of breakpoints use calibrated MIA advantage.

At `efConstruction` = 100 on MS-MARCO the breakpoint is $k = 128$, 192, 384, 1,024, and 3,200 for $n = 500$, 1,000, 2,000, 5,000, and 10,000, or 0.256, 0.192, 0.192, 0.205, and 0.320 as fractions of the database. That fraction shows no downward trend, so over the sizes tested this attack requires between a fifth and a third of the database. We report the advantage, acceptance rates, and AUC at each breakpoint, together with the point below it, in Table 4. We report measured breakpoints rather than fit a growth rate because five sizes over one order of magnitude do not determine one.

On data uniform on the sphere, the fraction decreases instead: the corresponding k values are 64, 96, 128, 224, and 512, or 0.128, 0.096, 0.064, 0.045, and 0.051. We do not interpret the last value as a reversal, since the matched attack is substantially stronger on the sphere at every size. We omit the $n = 200$ index because it uses `efConstruction` = 30.

The null. Because the threshold is selected after seeing the scores, empirical KS separation is positive even when the member and nonmember score distributions are identical. With finitely many runs, the two empirical distributions differ at some threshold, and the statistic takes the largest such difference. We therefore compare the reported separations against a *paired label-swap null*, obtained by exchanging the member and nonmember labels within randomly chosen pairs and recomputing the statistic. Only separation beyond the relevant null reference is evidence that the score distinguishes membership.

Unmatched attack results. At $n = 200$, the unmatched attack separation is 0.018, 0.014, and 0.013, beside an independent-sample reference of 0.017. That reference is not the exact null for paired scores, so the more direct control here is known-vertex recovery, which is 0.161, 0.162, and 0.160 against the random-set value $k/n = 0.160$. The beam search therefore does not place the known vertices usefully better than random selection. The larger sphere and MS-MARCO indexes behave the same way. At $k = 256$, the separations are 0.050 ± 0.007 on the sphere and 0.054 ± 0.004 on MS-MARCO, against paired-label-swap null 95th-percentile references of 0.086 and 0.085, and recovery is 0.051 on both, matching $k/n = 0.0512$. At $k = 512$ on MS-MARCO, recovery is 0.103 against 0.1024.

These results show that our edge-based placement strategy fails to extract a useful record-to-vertex assignment from the tested HNSW snapshots. They do not show that such an assignment is information-theoretically absent or that HNSW is secure in the unmatched setting. Stronger attacks could exploit global graph structure, higher-order topology, or a more faithful joint model of HNSW construction rather than the pairwise edge model used here.

The contrast with RAP helps explain the failure of this particular strategy. In RAP, every pair of records exposes a noisy numerical inner product. In HNSW, by design, only a sparse subset of geometric relationships survives as graph edges, and absence of an edge generally says little about the underlying distance. The final graph is also a history-dependent compression of the geometry: an edge depends on the construction search and neighbor-pruning steps, not only on the distance between its endpoints. Thus the known plaintexts do not directly provide a second copy of the leaked graph that can simply be aligned to it.

We also checked that the observed failure is not an artifact of the tested beam width. Widening the beam to 2,048 at $k = 128$ and 256 does not push the separation above its null. On the sphere at $k = 256$, where the matched attack exceeds separation 0.5, the unmatched attack reaches 0.092 ± 0.007 against a null 95th-percentile reference of 0.119. Comparing each seed against its own null rather than the seed-averaged reference puts three of the 45 completed seed–parameter

combinations above their 95th percentile, close to the roughly two expected under a no-information null.

F Experimental Setup and Additional Results

This appendix gives the setup shared by all our experiments, together with the results that no single section owns.

F.1 Experimental Setup

This appendix collects the run counts, seeds, beam widths, and reference-sample sizes for the attack experiments; utility and overhead settings are given where those results appear. The metrics we report are defined in Section 4.3.

Why we take the absolute value once. We average the signed acceptance-rate difference over seeds and take the absolute value at the end. Taking it per seed and then averaging would report positive advantage even for a score with no information, since each seed's difference is nonzero by chance. It would also let the attack choose the direction of its decision rule after seeing the test data, which the calibration runs prevent.

Retrieval metrics defined outside the body. Normalized discounted cumulative gain, nDCG@10, rewards placing relevant documents near the top, discounting each result by its rank and dividing by the best achievable value. Recall@100 is the fraction of relevant documents in the first 100 results. Both are computed against human relevance judgments rather than against plaintext search.

SAP **(Section 5.2).** Each point averages 30 independent hashed databases, each with a fresh sample. Each hashed database is scored on 100 held-out challenges balanced between members and nonmembers, so a point rests on 1,500 member and 1,500 nonmember challenges, and the smallest nonzero nonmember acceptance rate it can resolve is 1/1500. A separate calibration database of the same size fixes the score direction and threshold. We report standard errors across databases rather than across challenges, since challenges within a database share a scaling factor and are therefore not independent. We embed 24,000 MS-MARCO passages with `all-MiniLM-L6-v2` and normalize every embedding to unit length. The vectors are divided at random into disjoint public-reference, calibration, and test pools. Each test challenge database draws its vectors from the test pool, and nonmember challenges come from the unused ones. The default public reference has 2,000 vectors, and the remaining default parameters are $n = 2{,}000$, $d = 384$, and $\beta = 1$. The reference-size panel uses one public reference sample per size.

Methodology. Each held-out run uses a fresh sample, fresh perturbation coins, a fresh independent scaling factor ρ, and challenges balanced between members and nonmembers. The SAP key is the scaling factor ρ, which in these experiments is sampled independently from $[0.5, 2]$ for every calibration and held-out database. Labeled challenges from a separate calibration database select the score direction and threshold, which are then held fixed. We also report the fixed choice $\varepsilon = 0.5$. Error bars show one standard error across databases. Getr receives the specified variance for synthetic data and estimates it from a disjoint public reference set for MS-MARCO.

Data and parameters. The default synthetic setting is $n = 2{,}000$, $d = 10$, and $\beta = 1$, with independent samples from $\mathcal{N}(0, I_d)$ used for both the database and the nonmember challenges. Each sweep varies one parameter while holding the others fixed.

RAP, matched and unmatched (Section 6). We use beam width $B = 5000$. The matched attack uses five seeds, with 50 calibration runs and 100 held-out runs per seed, each run giving one member and one nonmember score. The sweep of Figure 6 uses three seeds, with 50 calibration and 100 held-out runs per seed for the k-sweep and 30 and 70 for the noise sweep. We measured the $n = 100$ unmatched results quoted in Section 6.6 with five seeds and a fresh key drawn for every run, with 30 calibration and 70 held-out runs per seed, in which every game samples a fresh key.

The matched CLIP/MS COCO check at $n = 100$ and $\beta = 0.5$ uses five seeds, 50 calibration games, and 100 held-out games per seed. The matched FaceNet/LFW experiment uses the same counts at every $k = 1, \ldots, 10$ for $n = 100$ and $n = 1{,}000$. Each game samples a fresh database, perturbation, and known-plaintext set. We select the lexicographically first funneled LFW image for each of 5,749 identities and encode it as a unit-normalized 512-dimensional FaceNet embedding using VGGFace2-pretrained weights. Thus every database/nonmember split is identity-disjoint by construction. The reported game is membership of the selected image's embedding, not identity membership from a different image.

Projected RAP (Section 7). Each setting uses five seeds and beam width $B = 5000$, with 30 calibration and 70 paired held-out runs per seed. Every game samples a fresh secret projection, database, noise, and shuffling permutation. Retrieval quality is measured separately on databases of size 10,000, where for each of three seeds 200 queries are searched using the plaintext and projected distances.

Comparison-only leakage (Section 8). Beam width is $B = 5000$. For each of five seeds, 30 calibration runs select the prefix width and threshold, and 70 paired held-out runs estimate advantage. Rank predictions use a public reference sample of 2,048 vectors disjoint from every challenge database. We average the signed member/nonmember acceptance-rate gap across the five seeds, take the absolute value, and use the standard error of those signed gaps for the error bars.

Encrypted HNSW index state (Section 9). At $n = 200$, each of four seeds fits the edge-probability curves on 500 independent databases and then runs 5,000 paired held-out runs with beam width $B = 512$. At the larger sizes the matched attack uses four seeds, 100 calibration runs, and 2,000 paired held-out runs per seed. Beam search is more expensive there, so the unmatched attack uses $B = 512$, four seeds, 50 calibration runs, and 500 paired held-out runs per seed. The calibrated sweep in Table 4 covers 121 settings. It runs MS-MARCO and uniform spherical data at $n = 500$, 1,000, 2,000, 5,000 and 10,000, and SciFact at $n = 2{,}000$, with between eight and fourteen values of k at each size. Every setting uses five seeds. For each seed, 100 separate public indexes fit the edge model, 50 paired calibration games fix the score direction and threshold, and 100 fresh paired held-out games estimate advantage, member and nonmember acceptance, and AUC. Each game samples a fresh database, index, insertion order, and known-plaintext set. The calibration games and the held-out games are disjoint, so no threshold is fixed on the games it is then scored on. Within one size and seed, the known-plaintext sets used for calibration are nested as k grows. The score direction is chosen separately for each seed and is not the same for all of them, so we average the signed held-out gap between member and nonmember acceptance across the five seeds and take the absolute value once at the end. Standard errors are taken across the five signed seed gaps.

Low false-positive rates. Carlini et al. [CCN+22] argue that a membership-inference attack should be evaluated at low false-positive rates rather than on average, and evaluate trained-model attacks that way. We do not report TPR at a prescribed low FPR, because our runs cannot resolve one. Each calibrated attack is evaluated on a few hundred to a few thousand held-out runs, so the smallest nonzero nonmember acceptance rate we can observe is between one part in a few hundred and one part in a few thousand, and at that level the standard error is as large as the estimate. A figure quoted at an FPR of 10^{-3} would be noise. Our experiments therefore measure average-case discrimination, and they do not rule out attacks that succeed at very low FPR. Our bounds do constrain low FPR values, since they hold at every threshold.

Embedding builds. Every comparison in the paper uses one embedding build on both sides.

F.2 Per-Record Membership Risk

Average attack performance can hide a minority of records that are far more exposed than the rest [CCN+22]. This appendix tests for such a minority in the matched known-plaintext attack of Section 6.5, on MS-MARCO at $n = 100$ and $\beta = 0.5$, with a fresh key drawn for every run and the same calibration split and seed count as the runs reported there. We reproduce the breakpoint reported in Section 6.5 at $k = 3$, so the quintiles below decompose the same measurement rather than a separate one.

We sort challenges by three properties of the challenge vector, each computed by us for the analysis and none of them available to the adversary. The *isolation* of a challenge is the Euclidean distance from it to the nearest other record in the database. Its *local density* is the number of database records within radius r of it, where r is the median nearest-neighbor distance over the database. Its *atypicality* is the Euclidean distance from it to the mean of the public reference set. For each property we split the challenges into equal-count quintiles by stable sorted order and report advantage using each run's calibration-fixed threshold. Equal-valued local-density records can therefore be divided across adjacent quintiles; those bins are not distinct value strata. Every cell uses 200 held-out challenges drawn from five seeds with 50 calibration and 100 held-out games each. We report the results in Table 10 at three values of k bracketing the breakpoint. Advantage rises with k. Within this experiment, none of the three properties has a monotone association with advantage; the largest within-property quintile gap is 0.182 for atypicality at $k = 3$. This descriptive analysis of one attack and dataset does not rule out concentration by these or other record properties, and it is not a repeated per-record risk estimate.

Adversarially chosen known plaintexts. The membership-inference game samples the k known plaintexts together with the rest of the database, so the adversary does not choose which records it knows. An adversary that chose them might do better. We tested this in a separate experiment, outside the game, using three rules for picking the k records. *Max-volume* adds at each step the record whose residual after projection onto the span of those already picked has the largest norm. *Max-leverage* whitens by the covariance of the public reference set and picks the k records with the largest whitened norm. *Cluster-cover* partitions the database into k clusters and picks the record nearest each centroid. Each rule uses only the database records the adversary is allowed to know and the public reference set; none of them uses the key, the hash values, or the challenge. We ran the matched attack of Section 6.5 against RAP on MS-MARCO at $\beta = 0.5$ for $k = 1, \ldots, 10$, at $n = 100$ and $n = 500$, with five seeds and the same calibration and held-out counts as above. All three rules give the same breakpoint as random sampling: $k = 3$ at $n = 100$ and $k = 4$ at $n = 500$. Max-leverage is lower than random at the breakpoint (0.562 versus 0.658 and 0.630 versus

Property	Quintile	Advantage (standard error)		
		$k = 2$	$k = 3$	$k = 4$
Isolation	1	0.280 (0.065)	0.700 (0.049)	0.903 (0.030)
	2	0.230 (0.065)	0.640 (0.054)	0.923 (0.026)
	3	0.244 (0.063)	0.640 (0.053)	0.940 (0.024)
	4	0.376 (0.064)	0.613 (0.056)	0.889 (0.035)
	5	0.308 (0.066)	0.698 (0.050)	0.861 (0.036)
Local density	1	0.330 (0.066)	0.685 (0.049)	0.940 (0.024)
	2	0.256 (0.063)	0.609 (0.057)	0.880 (0.033)
	3	0.263 (0.067)	0.661 (0.051)	0.890 (0.035)
	4	0.301 (0.064)	0.698 (0.049)	0.890 (0.032)
	5	0.267 (0.067)	0.644 (0.053)	0.913 (0.028)
Atypicality	1	0.230 (0.066)	0.659 (0.053)	0.889 (0.033)
	2	0.350 (0.065)	0.662 (0.053)	0.890 (0.032)
	3	0.247 (0.065)	0.581 (0.057)	0.898 (0.031)
	4	0.303 (0.066)	0.614 (0.054)	0.933 (0.025)
	5	0.296 (0.066)	0.763 (0.045)	0.899 (0.031)

Table 10: Advantage of the matched known-plaintext attack within quintiles of three properties of the challenge vector, at three values of the number k of known plaintexts. Each cell uses 200 held-out challenges and each run's own calibration-fixed threshold.

0.722, respectively), but it is not lower at every tested k. Under these three selection rules and two database sizes, the breakpoint is unchanged.

F.3 Performance and Scalability

A scheme in our context is useful only if it can run at the scale of the database it protects. This appendix is an implementation-overhead study on the CPU benchmarks, not a deployment-scale validation. Ordinary RAP does not change the size of a stored vector; measured batched query throughput falls by a factor of about two to three. These measurements are not controlled against protocols that replace the search backend. RAP does not change the search procedure, so a query moves no more data than it would without protection. The oblivious construction of Boldyreva and Tang [BT21], by contrast, moves about 44 MB between client and server for a single lookup in their protocol on a million-vector database, and a query makes ten such lookups.

Cost of hashing. Hashing one vector requires a $d \times d$ matrix–vector product and bounded noise, with the matrix product using about $2d^2$ floating-point operations, or 2.95×10^5 operations at $d = 384$. For context, a simple arithmetic estimate for a 22.7-million-parameter sentence encoder on 256 tokens is 1.16×10^{10} operations. The rotation multiply is therefore 0.0025% of this FLOP estimate; this is not an end-to-end timing comparison. The operation-count estimate and the separate hash microbenchmark are in Figure 9.

On an Apple M1 Pro with the linear-algebra threads set to one, our batched, unoptimized implementation hashes 0.35 million vectors per second at $d = 384$, including noise, and therefore takes about 29 seconds for ten million vectors. The rotation alone is faster because bounded-noise sampling dominates this implementation, while a single query hash takes about 14 microseconds, or 2.9 microseconds per query in a batch of 1000. The same operation measured inside the HNSW

benchmark below, on the Xeon and in batches of 500, costs about 24 microseconds per query, so we use the in-benchmark figure when accounting for query throughput.

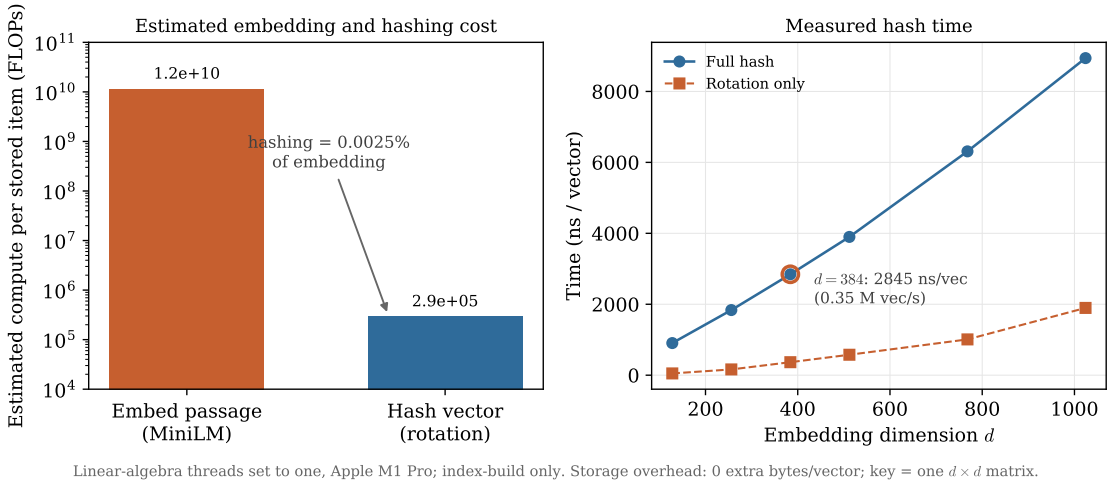

Figure 9: Cost of RAP. Left: estimated operation count, on a log scale, for embedding a 256-token passage with MiniLM and for the RAP rotation at $d = 384$. Right: measured batched hash time on an Apple M1 Pro with linear-algebra threads set to one. "Full hash" includes our bounded-noise sampler; "rotation only" isolates the matrix product.

Storage. If the hash values replace the plaintext embeddings, ordinary RAP uses the same storage, 1536 bytes per vector at $d = 384$, and its key is one $d \times d$ matrix, about 0.59 MB in 32-bit floating point. Ten million vectors take 15.36 GB in either representation, and projected RAP at $m = 192$ halves that to 7.68 GB. The index is larger than the vectors it holds, about 1808 bytes per vector at $d = 384$ in Table 11. The graph overhead depends mainly on the connection count, while the serialized index also holds a copy of the vectors, whose size depends on the dimension. Projection therefore shrinks the index by less than it shrinks the vectors, from 36.16 to 20.80 MB on MS-MARCO. In every case retrieval uses the same distance and the same type of index as it would on plaintext.

Retrieval agreement under exact search. We next check how often hashed retrieval returns the same top-10 neighbors as plaintext retrieval on the SciFact embeddings. The overlap is 1 without noise, as expected from an isometry, and 0.913 at $\beta = 0.5$. Secret projection lowers it further, to 0.662 at $m = 192$ and 0.535 at $m = 128$. We report task-level quality for these representations in Appendix C.4.

HNSW build and query benchmark. We also build and query Faiss HNSW indexes on plaintext and hashed vectors. The MS-MARCO experiment uses 20,000 database vectors and 500 queries, while SciFact uses its 5,183-document corpus and 300 test queries. We set `M` = 32, use `efConstruction` = 100 and `efSearch` = 128, and average three seeds on an Intel Xeon Gold 5220 using eight CPU threads. The task-aware row applies a public reduction from 384 to 192 dimensions and then a secret projection to 96.

The table reports two timings, both of which cover hashing one query and then searching the index. Query p50 is the median time for a single query, issued one at a time. Batch QPS is the

number of queries answered per second when the whole batch is issued at once. The fixed cost of making one call is a large share of the p50 figure, so p50 does not decompose the batch figures and the two columns need not order the rows the same way.

Retrieval quality can decrease here for two reasons, the secret-key hashing scheme and the approximation the index supports. We report a metric of each in Table 11. Both compare one retrieval against another, so despite the shared name neither is the Recall@k of Section 2, which is scored against the dataset's relevance judgments. *Index Recall@10* compares HNSW against exact search on the same vectors, and so isolates the approximation the index makes. *End-to-end Recall@10* compares HNSW on the hashed vectors against exact search on the plaintext vectors, and so combines hashing, projection, noise, and the index. We find that the index is not the source of the loss in retrieval quality. End-to-end Recall@10 falls from 1.000 to 0.498 across the rows, while Index Recall@10 stays at 0.998 or above, so nearly all discrepancy with plaintext search comes from the hashing scheme.

Data	Representation	Dimension	Build (s)	Index (MB)
MS-MARCO	Plaintext	384	1.84	36.16
	RAP	384	1.28	36.16
	Projected RAP	192	0.92	20.80
SciFact	Plaintext	384	0.15	9.37
	RAP	384	0.16	9.37
	Projected RAP	128	0.14	4.06
	Task-aware RAP	96	0.15	3.40

Data	Representation	Index R@10	End-to-end R@10	Query p50 (ms)	Batch QPS
MS-MARCO	Plaintext	1.000	1.000	0.57	10,880
	RAP	1.000	0.936	0.50	5,653
	Projected RAP	0.999	0.743	0.31	9,397
SciFact	Plaintext	0.998	0.998	0.17	40,383
	RAP	0.998	0.906	0.38	15,138
	Projected RAP	0.998	0.547	0.27	13,921
	Task-aware RAP	0.999	0.498	0.23	14,368

Table 11: End-to-end HNSW retrieval at $\beta = 0.5$ for the hashed representations. Times describe this implementation and hardware rather than performance guarantees.

Where the throughput cost comes from. Our benchmark measures the time for query hashing and index search separately. On MS-MARCO at `efSearch` = 128, hashing a query costs 23.7 microseconds in a batch, and searching the RAP index costs 147.7 microseconds against 91.9 for plaintext. On SciFact the same three figures are 13.2, 52.9, and 24.7. About a third of the gap is the hashing itself. The rest is search over the hashed vectors, which costs 1.6 times what it costs over plaintext on MS-MARCO, and 2.1 times on SciFact.

Comparison to access-controlled IPFRE. Acharya et al. [AFLO25] implement their complete access-controlled IPFRE protocol, a variant of the static FH-IPFE of Kim et al. [KLM$^+$18] that trades leakage against efficiency differently, built from their TinyFHIPFE scheme, on four datasets of 60,000 to 1.18 million vectors. On an Intel i7-9750H laptop, encryption takes 3.3–4.0 milliseconds per vector, a query takes 7.3–23.5 seconds, and an update takes 6.6–22.2 seconds. In our measurements

RAP hashes a database vector in about 2.9 microseconds when batched on the M1 Pro, and hashing plus HNSW search on the Xeon answers a query in 0.50 milliseconds for RAP and 0.31 for projected RAP on MS-MARCO.

Storage further separates these two measured configurations. AFLO report a TinyFHIPFE ciphertext of $d + 4$ group elements. Assuming 32-byte compressed source-group elements for their Curve25519 instantiation, $d = 384$ gives 12,416 bytes, versus 1536 bytes for one RAP vector, a factor of about eight. Extrapolated to ten million vectors, the stored vectors take 124 GB and 15.36 GB. The workloads and hardware are not controlled, so these figures compare the reported configurations rather than establish a class-wide performance ordering.

Task-aware retrieval quality after indexing. We separately repeated the SciFact quality measurement over 20 seeds at the two comparable projection-error settings used in Appendix C.4. After HNSW indexing, task-aware RAP at $384 \to 192 \to 96$ has nDCG@10 0.577 and MRR@10 0.546, compared with 0.571 and 0.532 for ordinary projected RAP at $384 \to 128$. Its Recall@100 is lower, 0.861 rather than 0.872, so the learned public map improves the top of the ranking in this experiment without improving every retrieval metric. Under exact search on the same twenty seeds the values are 0.578, 0.546, and 0.861 for task-aware RAP and 0.571, 0.533, and 0.873 for projected RAP; we measured the four-seed exact-search numbers in Table 7 separately. For each of these metrics, HNSW changes the exact-search value by at most 0.001.

A larger index on data uniform on the sphere. To test a larger database, we also index 100,000 vectors uniform on the sphere. The 384-dimensional plaintext index is 181 MB and takes 21 seconds to build, while projection to 192 dimensions reduces these figures to 104 MB and 6.9 seconds. Index Recall@10 on the plaintext sphere is only 0.291 at `efSearch` = 128 and 0.459 at 256, showing that uniformly random high-dimensional vectors are a difficult ANN workload even before hashing.