Exemem: Privacy-Preserving Network Effects on Vectorized Data

Tom Tang

March 4, 2026

Abstract

Exemem is a networked computation platform over vectorized data, built on Fold DB. Each entry in the network—an embedding plus metadata—is publicly exposed and tagged with a unique *derived pseudonym*: a one-time public key generated from the owner's master key. The owner can verify any entry is theirs, but the public cannot link two entries to the same person. Anyone can search across all embeddings and discover similarity matches, but results reveal only unlinkable per-entry pseudonyms—never real identities, and never which entries share an owner. This architecture creates powerful network effects: the more users who participate, the more complete the picture of where your data exists. A user can discover that 14 entries contain photos matching their face, each under a distinct pseudonym, without learning who owns any of them or whether they belong to one person or fourteen. We analyze the properties of this model, demonstrate applications from personal media to collective intelligence, and show that network growth strengthens anonymity rather than eroding it.

1 Introduction

Machine learning has made it practical to convert any data—photos, audio, text, documents, biometric scans—into high-dimensional vector embeddings. Two vectors close in embedding space represent semantically similar data: two photos of the same face, two documents on the same topic, two voice recordings of the same speaker.

If many users store embeddings of their data in a shared network, a new class of query becomes possible: *similarity discovery across users*. You can ask "who has data that looks like mine?" and get real answers. But answering this question naively—with real identities attached to results—creates surveillance infrastructure. The cure is worse than the disease.

Exemem takes a different approach. Every entry is tagged with a unique **derived pseudonym**—a one-time public key generated from the owner's master key. No two entries share a pseudonym, even if they belong to the same person. Each entry's metadata—including its vector embedding—is **publicly exposed**. There are no access gates on the embeddings themselves. Anyone can search the full embedding space, find matches, and see each match's derived pseudonym. The privacy guarantee is not that the data is hidden, but that **no two entries can be linked to the same owner, and no entry can be linked to a real identity**.

This is a fundamentally different privacy model from Fold DB's fold-based access control [1], which protects raw data behind policy-enforcing interfaces. The two work in concert: Exemem exposes embeddings and metadata for public discovery; Fold DB protects the underlying raw files.

This paper makes three contributions:

1. A model for *per-entry pseudonymous networks*: public embeddings, public metadata, unlinkable derived pseudonyms per entry, with raw data behind folds.

2. A *similarity discovery* protocol where results include per-entry pseudonyms that cannot be linked to each other or to real identities.
3. An analysis of *network effects*: value grows with participation, and anonymity strengthens rather than weakens as the network grows.

2 Architecture

2.1 Two Layers

The system has two distinct layers:

Public layer: embeddings and metadata. Every user's vector embeddings and file metadata (media type, timestamp, embedding transform hash) are publicly queryable. The transform hash identifies which UTR transform produced the embedding, enabling cross-entry similarity comparisons; it also means an adversary who knows the model architecture can attempt model-specific inversion attacks (see Section **??**). There are no access controls on this layer. Anyone can search, compare, and discover matches. Each entry is tagged with a unique derived pseudonym—a one-time public key that cannot be linked to the user's other entries or to their real identity.

Private layer: raw data behind folds. The actual files—photos, documents, recordings—are protected by Fold DB [1]. Access to raw data requires passing the fold's policy checks (trust distance, cryptographic capabilities, payment). The details of fold-based access control are described in [1].

The public layer enables discovery. The private layer protects content. A querier can learn that an entry with derived pseudonym `0xA7f3...` contains a matching face embedding, but accessing the actual photo requires passing the fold's policies. They cannot determine whether `0xA7f3...` and another matching entry `0x91cD...` belong to the same person.

2.2 Per-Entry Pseudonymous Identity

Definition 1 (Derived Pseudonym)**.** *Each entry in the public layer is tagged with a unique* derived pseudonym*—a public key deterministically generated from the user's master private key and a per-entry index (e.g.,* $pk_i = \text{Derive}(sk, i)$*, using a hierarchical deterministic key derivation scheme such as BIP-32 or HMAC-based derivation on an elliptic curve). The required security property is that* $\text{Derive}(sk, i)$ *and* $\text{Derive}(sk, j)$ *are computationally indistinguishable from two independently generated keys to any party that does not know* sk*. The user can verify that any derived pseudonym is theirs (by re-deriving it from their private key), but the public cannot link two derived pseudonyms to the same user. Every entry appears to come from a distinct, unrelated identity.*

This is stronger than conventional pseudonymity, where a single pseudonym is reused across entries. With per-entry derivation:

- An observer who sees 1,000 entries sees 1,000 unrelated pseudonyms. They cannot determine whether these belong to 1,000 users or 1 user.
- Correlation attacks based on grouping entries by pseudonym are impossible—there is nothing to group.
- The user retains full ownership: given their master key, they can prove authorship of any entry or re-derive any pseudonym on demand.

2.3 Embeddings as Public Metadata

When a user stores a file, the system generates vector embeddings using a registered transform T_{embed} from the Universal Transform Registry:

$$T_{\text{embed}} : \text{RawData} \to \mathbb{R}^d$$

The embedding is practically irreversible: exact reconstruction of the raw data from the vector is infeasible, though approximate reconstruction may be possible in some cases (see Section **??**). Because all users who reference the same T_{embed} hash produce vectors in the same embedding space, similarity comparisons are valid across the entire network.

The embedding, along with non-sensitive metadata, is published to the public layer. The raw file goes behind the user's fold. This split is the key design decision: embeddings are public because they must be searchable; raw data is private because it must be protectable.

3 Similarity Discovery

3.1 The Protocol

A user submits a query vector $q \in \mathbb{R}^d$ and a similarity threshold ϵ. The system searches the public embedding index:

$$\text{Result}(q, \epsilon) = \{(pk_i,\ \text{sim}(q, e_i),\ m_i) : \text{sim}(q, e_i) \geq \epsilon\}$$

The result is a set of triples: the entry's derived pseudonym, the similarity score, and the public metadata. Every match is returned—there is no per-user policy check at this layer, because the embeddings are public.

Query privacy. The query vector q itself is sensitive: a face-embedding query reveals the querier's face. The system should not require the querier to identify themselves. Queries can be submitted pseudonymously (using a derived pseudonym) or over an anonymizing transport (Tor, VPN). The index server sees the query vector but not who submitted it. Stronger protection—where even the index server cannot see q—requires encrypted similarity computation (Section **??**), which is future work.

The querier can see:

- A unique pseudonym for each matching entry.
- The similarity score and metadata for each match.

The querier *cannot* see:

- Who owns any entry (the derived pseudonym is unlinkable to a real identity).
- Whether any two entries belong to the same person (each has a distinct derived pseudonym).
- The raw data behind the embedding (that requires fold access).

3.2 From Discovery to Access

Discovery and access are separate operations with separate trust models:

1. **Discovery** (public layer): Alice searches for face embeddings similar to hers. She finds 14 matching entries, each with a unique derived pseudonym. She cannot tell if they belong to 14 people or 1 person.

2. **Access request** (private layer): Alice requests access to the raw photo behind a specific entry's derived pseudonym. The request is routed to the owner's Fold DB instance, which applies its access policies. If she passes, she sees the photo. If not, she gets nothing.

The owner controls raw-data access through Fold DB's policy system. The public embedding layer gives Alice the ability to *find* the data; Fold DB gives the owner the ability to *control* it.

4 The Photo Discovery Problem

4.1 Scenario

Consider the question: *"Do strangers have photos of me?"* Today, answering this requires either centralized facial recognition databases (privacy-destroying at scale) or asking every person individually (impractical).

4.2 Setup

A face-embedding transform T_{face} is registered in the UTR, producing 512-dimensional vectors. Users who store photos opt into the face-embedding schema: each photo is processed, faces are detected and embedded, and the embeddings are published to the public layer with metadata (media type, capture date). The raw photos remain behind each user's fold.

Note the consent asymmetry: the data *owner* opts into publishing embeddings, but the data *subjects* (people whose faces appear in the photos) do not consent to being embedded. Exemem does not solve this problem—it is inherent to any system that processes photos containing other people's faces. What Exemem does provide is the *discovery* mechanism that lets subjects find out they've been embedded, which is strictly better than the status quo where subjects have no way to know. The removal-request protocol (Section **??**) gives subjects a channel to act on that knowledge.

4.3 Discovery

Alice wants to know if anyone has photos of her:

1. She generates her own face embedding q_{Alice} using the same T_{face}.

2. She queries the public index: $\text{sim}(q_{\text{Alice}}, \cdot) \geq 0.9$.

3. She receives results:

Derived Pseudonym	Similarity	Type	Date
`0xA7f3...`	0.98	photo	2025-01
`0xB2e1...`	0.91	video frame	2025-06
`0x91cD...`	0.95	photo	2024-08
`0x3eF0...`	0.93	photo	2024-03

... 10 more entries ...

Alice now knows: 14 images containing her face exist in the network. She has a similarity score, media type, and date for each.

Alice does *not* know:

- Who owns any of these images—each entry has a unique derived pseudonym.
- Whether any two entries belong to the same person. Three photos from the same person look like three unrelated pseudonyms.
- What the photos look like or where they were taken.

4.4 Responding to Discovery

Requesting access. Alice can request access to the raw photo behind any entry's derived pseudonym. The request is routed to the owner's Fold DB instance. One owner might allow broad access; another might deny all access. Alice sees only the photos she's authorized to see. Even after access, she cannot link two permissive entries to the same owner—the derived pseudonyms are independent.

Requesting removal. Alice can send a signed message to any matching derived pseudonym: "the subject of face embedding q requests removal." The message is routed without revealing Alice's identity (she uses her own derived pseudonym for the request). The owner sees the request; whether they act on it is their decision, but the request and its timestamp are recorded in the append-only store. The system provides *awareness and auditability*, not enforcement—there is no mechanism to force an owner to delete their data.

Monitoring over time. Alice can re-run her query periodically. If new entries appear with matching embeddings, she is aware. Note that the embedding layer is append-only: even if an owner removes the raw file from their fold, the embedding record persists—it documents that the data once existed. This means a removal request, even if honored at the fold layer, does not erase the embedding. This is a deliberate trade-off: the append-only record provides accountability (the owner cannot deny the data ever existed), at the cost of permanent discoverability at the embedding level. This design is in tension with right-to-erasure regulations such as GDPR Article 17. Deployments in jurisdictions with such requirements may need a tombstoning mechanism that marks embeddings as withdrawn (preventing them from appearing in search results) while retaining the record for audit purposes. The design of such a mechanism—balancing erasure compliance with append-only accountability—is future work.

5 Network Effects

5.1 Value Growth

The value of the network grows with participation along two dimensions.

Discovery value. Each additional user increases the probability of finding relevant matches. Under an independence assumption, for a query with true match rate p per user, the probability of at least one match is $1 - (1 - p)^n$, approaching 1 as n grows. In practice, match rates are not independent—social connections increase co-occurrence (friends are more likely to have photos of each other)—but the qualitative conclusion holds: more users means more complete answers.

Coverage value. A network with 1,000 users captures a small slice of existing data. A network with 100 million users captures a representative sample. Users with unique data (uncommon locations, rare content types) contribute disproportionate value.

5.2 Anonymity Strengthens with Growth

In traditional systems, network growth erodes privacy: more data, more profiling, more deanonymization vectors. In a per-entry pseudonymous network, the opposite occurs.

Observation (Anonymity Growth). Let N be the number of users and E the number of entries. For any derived pseudonym pk_i, the anonymity set—the set of users who could plausibly be the owner—is N, because pseudonyms are unlinkable and carry no information about their owner. An adversary who observes k matching entries cannot reduce the anonymity set by grouping them: each entry's pseudonym is independently derived, so the adversary's best guess for the owner of any single entry remains uniform over all N users. As N grows, the anonymity set grows linearly. Unlike formal anonymity definitions (e.g., k-anonymity or differential privacy), this is an architectural observation about the system's information-theoretic properties, not a proven bound against arbitrary side-channel attacks.

With 1,000 users and 100,000 entries, each entry could belong to any of 1,000 people, and no two entries can be grouped. With 1 million users and 100 million entries, the anonymity set is vastly larger. Network growth makes deanonymization harder, not easier—the exact opposite of conventional social networks.

5.3 Collective Computation

Because embeddings are public, deterministic programs can run across the entire network without per-user access negotiation. Examples:

- **Clustering**: group entries whose document embeddings are similar, revealing topic clusters without revealing content, identity, or whether clustered entries belong to the same person.
- **Anomaly detection**: identify entries whose embeddings deviate from the network norm, flagging unusual patterns without inspecting raw data or linking entries to users.
- **Aggregate statistics**: compute network-wide distributions ("what fraction of entries are face embeddings with similarity above 0.9 to a known reference?") from public data.

These computations are possible precisely because the embedding layer is public. No per-user policy negotiation is needed. The raw data remains private; the embeddings power collective intelligence.

6 Applications

6.1 Intellectual Property Discovery

Creators embed their work (text, audio, code) using registered transforms. The embeddings are public. A musician queries: "does anyone have audio similar to my track?" She sees an entry with derived pseudonym `0xF4a2...` at 0.96 similarity, uploaded 6 months before her release. She can't see who owns it or whether the same person has other entries. She can request access to the raw audio through the entry's fold, or file a timestamped claim referencing the embedding.

6.2 Medical Cohort Discovery

Patients publish clinical profile embeddings (symptom vectors, treatment response patterns), each under a unique derived pseudonym. A researcher queries: "how many entries match this rare disease signature?" She sees 47 matching entries. She cannot tell how many distinct patients they represent. She can contact any entry's derived pseudonym through the system to propose clinical trial participation—without knowing who they are unless they choose to reveal themselves.

Caveat: clinical embeddings that are close to a known rare-disease signature effectively reveal the diagnosis, even without raw data access. Patients publishing such embeddings should understand that the embedding itself leaks diagnostic information. Privacy-preserving embedding transforms (e.g., adding calibrated noise, reducing dimensionality) can reduce this risk at the cost of discovery precision.

6.3 Threat Intelligence

Security teams publish embeddings of indicators of compromise (malware signatures, traffic patterns), each entry under a unique derived pseudonym. A team queries: "has anyone seen patterns similar to this attack?" They see 7 matching entries. They can't tell which or how many organizations are affected, but they know the threat is widespread and can share defensive information through the derived pseudonyms.

6.4 Collective Knowledge Networks

Researchers publish embeddings of their findings, each under a unique derived pseudonym. Similarity queries reveal where knowledge overlaps. A group discovers 23 entries with highly similar research embeddings—potential signs of parallel work. They can reach out to each entry's derived pseudonym to explore collaboration before revealing identities, without knowing whether 23 entries represent 23 groups or 3.

7 System Considerations

7.1 Embedding Index

Because embeddings are public, the system can maintain a global vector index (e.g., using approximate nearest neighbor structures like HNSW or IVF). There is no per-user policy check at query time—the index serves all queries directly. This makes similarity search sublinear in the number of entries, rather than requiring $O(n)$ fold evaluations.

7.2 Embedding Inversion Risk

Neural network embeddings are not formally irreversible—there is no mathematical proof that the original data cannot be recovered. In practice, exact reconstruction is infeasible, but research has shown that approximate reconstruction is sometimes possible (e.g., generating a face that matches a face embedding). Since embeddings are public in this model, this risk is inherent. Mitigations include:

- Registering embedding transforms that are specifically designed to resist inversion (e.g., adding structured noise, reducing dimensionality).

- Accepting that embeddings reveal *some* information about the data (semantic similarity) while protecting the *specific* content (the actual photo, the exact text).
- Relying on per-entry pseudonymity: even if someone reconstructs an approximate face from an embedding, they cannot link the derived pseudonym to a real person or to any other entry.

7.3 Deanonymization Resistance

The primary deanonymization risk is linking a derived pseudonym to a real person using side information. Per-entry derivation eliminates the most common attack vector—correlating multiple entries by shared pseudonym—because no two entries share a pseudonym. Remaining defenses:

- **No side channels**: the system stores no IP addresses, no session data, no timing information. Connection-level privacy (Tor, VPN) is orthogonal but complementary.
- **Content-based correlation**: an adversary might attempt to link entries by embedding similarity (e.g., "these 50 photos were all taken in the same house"). This is an inherent limitation of publishing embeddings—semantic information is intentionally exposed. The mitigation is that such correlation yields probabilistic guesses, not definitive links, and the anonymity set remains large.
- **Sybil attacks**: because per-entry pseudonyms are free to generate, an attacker can flood the network with fake entries to poison similarity searches (causing queries to return attacker-controlled matches) or to dilute the anonymity set with entries the attacker controls. Mitigations include proof-of-work or proof-of-stake costs on entry publication, rate limiting per node, and reputation systems that weight results by entry age and access history. A full Sybil-resistance mechanism is outside the scope of this paper.

7.4 Interaction with Fold Access

Routing without linkability. When a querier requests access to raw data behind a derived pseudonym, the system must route the request to the owner without revealing the mapping from pseudonym to owner. This is achieved through an encrypted routing table: when an entry is published, the owner registers an encrypted forwarding record that maps the derived pseudonym to their node address, encrypted under the system's routing key. The routing service can forward requests to the correct node without learning which pseudonyms belong to the same owner, because each forwarding record is independent and reveals only a single pseudonym-to-node mapping per lookup. To prevent the routing service from linking pseudonyms by destination address, the owner may use per-entry relay addresses or an onion-routing layer. The design of a fully unlinkable routing protocol is an active area; the current model assumes a trusted routing service that does not collude with queriers.

Fold access. Once routed, the owner's Fold DB instance applies its policy checks [1]. If access is granted, the querier sees the raw data for that single entry but still cannot learn the owner's real identity or link this entry to any other entry—the fold returns data, not identity. Even granting access to multiple entries does not reveal whether they share an owner.

8 Related Work

- **Private information retrieval (PIR).** PIR protocols [2] allow querying a database without revealing the query. In Exemem, the query is not hidden (it is a publicly submitted vector), but the pseudonymous owner's real identity is.
- **Federated learning.** Federated learning [3] trains models across distributed data without centralizing it. The pseudonymous embedding network enables a different operation—discovery and search—with public embeddings and private raw data.
- **Decentralized identity.** Self-sovereign identity systems (DIDs, Verifiable Credentials) provide pseudonymous identity infrastructure. Exemem's pseudonymous model could be implemented on top of such systems, inheriting their key management and revocation properties.
- **Stealth addresses.** Monero and similar cryptocurrencies use per-transaction derived keys so that no two payments to the same recipient share an address. Exemem's per-entry derived pseudonyms apply the same principle to data entries rather than financial transactions.
- **Anonymity networks.** Tor and mix networks provide connection-level unlinkability—hiding who is communicating with whom. Exemem provides entry-level unlinkability—hiding which entries share an owner. The two are complementary: Tor protects the network layer, Exemem protects the data layer.
- **Content-addressable networks.** IPFS and similar systems identify data by content hash, enabling decentralized storage and retrieval. Exemem's public layer shares the principle of content-derived identifiers but applies it to semantic embeddings rather than exact content, enabling similarity search rather than exact lookup.
- **Reverse image search.** Centralized reverse image search (Google, TinEye) indexes public images with real-world attribution. Exemem extends this to all users' data with pseudonymous attribution and fold-gated raw access.

9 Future Work

- **Differential privacy on aggregates**: adding calibrated noise to aggregate statistics computed over public embeddings to prevent inference about individual users from network-level patterns.
- **Cross-modal discovery**: querying across modalities (text matched against image embeddings) using shared representation spaces, enabling discovery like "has anyone published images related to this text?"
- **Temporal discovery**: tracking how the number of matches for a query changes over time, enabling users to detect new appearances of their data in the network.
- **Reputation without linkability**: building trust signals for entry owners (quality of embeddings, responsiveness to access requests) without linking entries or revealing identity—potentially via zero-knowledge proofs of past good behavior.
- **Encrypted similarity**: computing similarity on encrypted embeddings so that even the index server cannot see the vectors in cleartext, providing defense in depth beyond pseudonymity.

10 Conclusion

Per-entry pseudonymous embedding networks create a new kind of collective intelligence: one where all data is discoverable but no two entries can be linked to the same person, and raw content remains behind policy-enforcing folds. The architecture has two layers—a public layer of embeddings tagged with per-entry derived pseudonyms for discovery and unlinkability, and a private layer powered by Fold DB for raw-data access control.

The key properties are:

- **Full discoverability**: every embedding is searchable by everyone. No access gates on discovery.
- **Per-entry unlinkability**: each entry has a unique derived pseudonym. No two entries can be linked to the same owner.
- **Owner verifiability**: despite unlinkability, the owner can prove authorship of any entry by re-deriving its pseudonym from their master key.
- **Protected content**: the raw data behind the embeddings is still controlled by folds with trust distance, capabilities, and payment.
- **Growing value**: more participants mean better discovery coverage.
- **Growing anonymity**: larger networks make deanonymization harder, not easier.

The result: the answer to "do strangers have photos of me?" is answerable—you see 14 matching entries, each under a distinct derived pseudonym—without learning who owns any of them, whether any share an owner, or creating the surveillance infrastructure that linking entries to real people would require.

References

[1] T. Tang, "Fold DB: Compute Without Exposure," 2026.

[2] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proc. 36th IEEE FOCS*, pp. 41–50, 1995.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th AISTATS*, pp. 1273–1282, 2017.