Embeddings Are Not Private

Attack evidence, layered defenses, and what you can actually deploy today

The Agile Monkeys

March 2026

There is a widespread assumption in the enterprise AI world that goes something like this: "We store embeddings, not the original text. The data is safe — you can't reverse an embedding back into words."

This assumption is wrong. Demonstrably, empirically, dangerously wrong.

Embedding inversion attacks can recover 92% of the original tokens from a 32-token input (Morris et al., Cornell, 2023). On clinical data, attackers have extracted patient ages with 98.84% accuracy and sex with 99.47% accuracy — from embeddings alone (Huang et al., ACL 2024). The latest zero-shot attack achieves 82-92% semantic information leakage across any embedding model without per-model training (Zhang et al., 2025). As the authors put it: **"From the security perspective, sharing the embeddings of confidential or sensitive documents with third-party services is equivalent to sharing the documents themselves."**

If your organization stores embeddings of sensitive data — customer records, internal communications, medical notes, financial documents — in a vector database without additional protections, that data is recoverable. Not theoretically. Practically. Using publicly available tools.

This article presents the evidence, explains why the most common defenses fail, and lays out what you can actually deploy today to protect embedding-based systems — along with what's coming from research that will strengthen these defenses further.

The Attack Landscape

Embedding inversion — recovering original text from its vector representation — has progressed through three generations of increasingly powerful attacks. Each generation lowered the barrier to execution while improving recovery rates.

Generation 1: vec2text (2023)

The foundational attack came from Morris et al. at Cornell University, published at EMNLP 2023. Their vec2text method uses an iterative correction approach: generate an initial text hypothesis, embed it, compare the resulting embedding to the target, and refine. Repeat until the generated embedding converges on the target.

The results were stark. On 32-token inputs, vec2text achieved:

- **92% exact token recovery**
- **97.3 BLEU score** (near-perfect reconstruction fidelity)

- Successful extraction of full patient names from clinical notes in the MIMIC dataset

The method requires query access to the embedding model — either white-box access or API access to generate embeddings for comparison. The code is publicly available on GitHub, making the attack accessible to anyone with moderate technical skill.

The authors themselves proposed Gaussian noise addition as a mitigation. As we'll see, this defense has since been defeated.

Generation 2: Transferable Embedding Inversion (2024)

Huang et al., published at ACL 2024, escalated the threat model dramatically. Their key innovation: **the attacker does not need access to the victim's embedding model.**

Instead, the attacker trains a surrogate model using as few as 4,000 leaked document-embedding pairs — a realistic scenario in any organization that has ever experienced a partial data breach. Using adversarial training, the surrogate becomes functionally indistinguishable from the target model.

On clinical data from MIMIC-III, the transferable attack recovered:

- **98.84% of patient ages**
- **99.47% of patient sex**
- A 40-50% improvement over standard (non-transferable) attacks

The implications are severe. A breach of a vector database doesn't just expose embeddings — it exposes a practical pathway to reconstruct the original data, provided the attacker can also obtain a small number of document-embedding pairs to train a surrogate model. The 4,000-pair requirement is small relative to the scale of enterprise vector databases.

Generation 3: ZSinvert (2025)

ZSinvert, from Zhang et al., represents the current state of the art — and the most alarming advance. It is a **universal, zero-shot** inversion method that requires no training of an inversion model per embedding type. The same algorithm works across different embedding architectures using adversarial decoding combined with a reusable offline correction model.

Where vec2text required substantial per-model training (the authors report multi-day training runs on multiple GPUs), ZSinvert works out of the box on any model.

On sensitive Enron email data, ZSinvert achieved **82-92% information leakage** as judged by an LLM evaluating whether reconstructions revealed key sensitive content. Critically, ZSinvert **maintains effectiveness**

against Gaussian noise defenses — the mitigation proposed by the original vec2text authors — undermining the most frequently discussed defense.

The authors' conclusion is unambiguous: "From the security perspective, sharing the embeddings of confidential or sensitive documents with third-party services is equivalent to sharing the documents themselves."

Practical Validation: It Works in the Real World

Academic attacks might seem theoretical. Tonic.ai, a data privacy company, validated these findings using production embedding models (OpenAI's text-embedding-ada-002) and the publicly available vec2text toolkit.

Their results bridge the gap between academia and practice:

- **40% of all sensitive data** in sentence-length embeddings is recoverable with exact match
- For longer texts (~5,000 characters), **over 10% remains exactly recoverable**
- Names, organizations, and geographic locations are substantially more recoverable than numeric data (phone numbers, credit cards) — a pattern that may reflect lower structural variability in text-based identifiers

This is not a lab experiment. These are production embeddings from the most widely used embedding API in the world, attacked with open-source tools.

The Trend Is Clear

Attack	Recovery Rate	Requirements	Year
vec2text	92% exact (32 tokens)	Query access to embedding model	2023
TEI (transferable)	98.84% attribute recovery	4,000 leaked pairs, no model access needed	2024
ZSinvert (zero-shot)	82-92% information leakage	Black-box access, no per-model training	2025
Tonic.ai (practical)	40% exact PII match	Publicly available tools	2024

Attacks are getting stronger, cheaper, and more general. The barrier to executing them is falling. Code is publicly available. Transfer attacks remove the need for model access. Zero-shot methods eliminate per-model training costs.

Every year, the attacker's job gets easier.

An Important Caveat: Model Coverage

An honest disclosure: the headline numbers above were measured on older, smaller embedding models — not on the latest generation. vec2text tested on GTR-base (768 dimensions) and text-embedding-ada-002 (1536 dimensions). ZSinvert tested on four open-source encoders (Contriever, GTE, GTE-Qwen2, GTR), none of them commercial APIs. Tonic.ai tested exclusively on ada-002. Huang et al.'s transferable attack tested on ada-002, SBERT, and Sentence-T5.

None of these papers tested on OpenAI's text-embedding-3-large (3072 dimensions), which is the current flagship embedding model for many enterprises.

The closest data point comes from the Eguard defense paper (2024), which did test on text-embedding-3-small, text-embedding-3-large, and ada-002. Their results showed undefended embeddings across all three models — including 3-large — achieved 93-98% F1 scores on inversion attacks using a GPT-2-based decoder. This suggests the vulnerability extends to newer models, but uses a different attack methodology than vec2text or ZSinvert.

A reproducibility study of vec2text (Zhuang et al., 2025) further qualified the original results: out-of-domain performance is "significantly lower" than reported, and the 92% recovery rate is specific to 32-token inputs — longer texts are substantially harder to invert exactly.

What does this mean? The core thesis — that embeddings encode recoverable semantic information — rests on sound principles that are unlikely to change with larger dimensions. More dimensions encode more information, not less. But the specific attack success rates cited in this article were measured on models one generation behind the current frontier. The risk is real; the exact percentages on text-embedding-3-large remain to be precisely quantified by the research community.

We present these numbers because they are the best available evidence, and because the trend is clearly toward stronger attacks, not weaker ones. But intellectual honesty requires noting where the evidence stops and extrapolation begins.

Why Naive Defenses Fail

Organizations that are aware of the inversion risk typically reach for one of four defenses. None of them is sufficient alone.

"Add Gaussian noise to the embeddings"

This was the original mitigation proposed alongside vec2text. The idea is sound in principle: perturb embeddings enough that inversion produces garbage, but not so much that similarity search breaks.

In practice, it fails on two fronts. First, ZSinvert (2025) explicitly maintains effectiveness against Gaussian noise defenses. The attack's adversarial decoding is robust to the noise levels that preserve useful retrieval quality. Second, the privacy-utility tradeoff is brutal: the amount of noise needed to meaningfully degrade inversion attacks also destroys retrieval performance. Uniform noise is a blunt instrument applied to a precision problem.

"Reduce embedding dimensionality"

Lower-dimensional embeddings carry less information, so inversion should be harder. This is true in theory, but insufficient in practice. Dimensionality reduction alone does not provide formal privacy guarantees, and the information loss hurts retrieval quality before it meaningfully hurts attackers. Attackers may still be able to invert lower-dimensional embeddings effectively, since the information that matters for PII recovery can be concentrated in relatively few dimensions.

"Put access controls on the vector database"

Access control on the vector database prevents unauthorized access to embeddings — which is necessary but not sufficient. It protects against external attackers but not against:

- Insider threats (database administrators, cloud provider employees)
- Database breaches (the embeddings are stored in plaintext)
- Third-party access (if you use a managed vector database service, the provider has access)
- Backup and log exposure (embeddings in backups, debug logs, or monitoring systems)

Access control is a perimeter defense. Embedding inversion is a data-at-rest attack. They address different threat models.

"Just don't store sensitive data"

In theory, the safest embedding is one generated from non-sensitive text. In practice, this is unrealistic for enterprise knowledge bases. The entire point of enterprise RAG is to make organizational knowledge — which inherently contains sensitive information — accessible to AI systems. Telling organizations not to embed sensitive data is telling them not to use the technology.

What You Can Deploy Today

The defenses above fail because they treat embedding privacy as a single-layer problem. A defensible architecture requires multiple layers, each addressing a different threat vector. The good news: three layers are deployable today using production-ready technology.

Layer 1: Pre-Embedding Sanitization

Maturity: Production-ready. Effectiveness: Drops PII recovery to zero.

The simplest and most effective defense is also the most overlooked: strip sensitive information from text before generating embeddings. If PII never enters the embedding, it cannot be recovered from it.

Tonic.ai validated this directly: when PII was redacted before embedding, recovery rates dropped to zero. This is not a theoretical result — it was tested on production embeddings.

For enterprise knowledge bases, pre-embedding sanitization means:

- Named entity recognition to identify and redact names, organizations, locations
- Pattern matching for structured data (credit card numbers, SSNs, phone numbers)
- Context-aware redaction that preserves semantic meaning ("Dr. Smith discussed the patient's cardiac condition" → "A physician discussed the patient's cardiac condition")

The challenge is balancing sanitization aggressiveness with semantic preservation. Too aggressive, and the embeddings lose the meaning that makes them useful. Too lenient, and identifiable context leaks through. For cross-organizational topic matching specifically, topic descriptions should be abstract enough that they capture the domain and problem type without revealing project-specific details.

Standard NER and PII detection tools (SpaCy, Presidio, commercial offerings) handle the straightforward cases. The harder problem — stripping identifying context while preserving useful semantics — remains an active area of work, but a well-designed sanitization pipeline significantly reduces the attack surface even without perfection.

Layer 2: Application-Layer Encryption

Maturity: Commercial product available. Effectiveness: Inverted embeddings produce nonsense.

IronCore Labs' Cloaked AI product encrypts embeddings at the application layer using the Scale and Perturb algorithm. Three mechanisms work

together: a scaling factor maps vectors to different ranges per encryption key, perturbation tweaks values while preserving approximate distance relationships, and element shuffling reorders embedding positions based on the key.

The result: encrypted embeddings produce "pure nonsense" when inverted. Critically, the encryption preserves approximate distance relationships, meaning vector similarity search still works on encrypted embeddings — but only for parties holding the encryption key. An approximation factor parameter controls the security-accuracy tradeoff.

This approach works with existing vector databases (Pinecone, Weaviate, Qdrant, pgvector) without modification. It doesn't require changes to embedding models or search infrastructure. It is less theoretically rigorous than full cryptographic approaches (see the research section below), but it is deployable today, commercially supported, and effective against current inversion attacks.

Layer 3: Bilateral Consent Protocol

Maturity: Architectural pattern. Effectiveness: Prevents unauthorized information sharing.

For cross-organizational intelligence — detecting that two teams are working on related problems — bilateral consent ensures that neither party learns anything about the other without explicit, mutual opt-in.

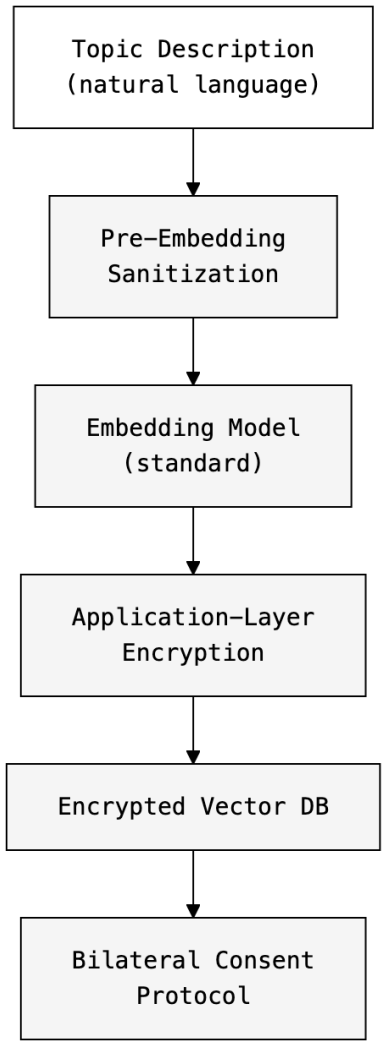
The protocol is straightforward:

- The system detects semantic similarity between two sanitized, encrypted topic embeddings
- Each party's agent notifies them: "Someone in Department Y is working on something potentially related to your project. Would you like to connect?"
- Only if **both** parties opt in does the system share contact information and sanitized topic descriptions
- If either declines, no information is exchanged — not even the fact that a match was detected

This is not a cryptographic defense — it's a process control. It's the last gate, not the primary privacy mechanism. But it's essential because it ensures that even if the technical layers have weaknesses, information sharing requires active human consent.

The Deployable Architecture

Combining these three layers:



This architecture is deployable today with existing tools. It does not require research prototypes. It will not stop a nation-state adversary with unlimited resources, but it raises the cost of attack well above the value of the information for realistic enterprise threat models.

Where Research Is Heading

The deployable architecture above provides meaningful protection, but stronger defenses are emerging from research. These are not yet production-ready, but they represent the trajectory of the field and may become deployable as they mature.

Eguard: Learned Projection Defense

Eguard, published in 2024, takes a fundamentally different approach to embedding privacy. Instead of adding noise (which degrades utility) or

encrypting (which requires key management), Eguard trains a projection network that minimizes the mutual information between original text and the output embedding while preserving downstream task utility.

The results are the best privacy-utility tradeoff in the literature: inversion attack F1 scores drop to **3.5-5.6%** (from 90%+ undefended), while downstream task accuracy remains high — 93-97% on sentiment classification, 96-98% on question retrieval, though results vary by model and task type. For comparison, standard differential privacy approaches (DPforward) achieve weaker privacy (F1 of 9-14%) with significantly worse task accuracy (46-75%).

The key insight is that privacy and utility don't have to be zero-sum. By learning which information to remove (the text-recoverable signal) rather than applying uniform noise, Eguard achieves dramatically better results on both dimensions.

Status: Research prototype. No public production implementation. The approach is architecturally simple (a single projection layer applied after embedding), which suggests relatively straightforward productization.

SPARSE: Concept-Aware Differential Privacy

SPARSE, currently under review at ICLR 2026, addresses the core limitation of uniform differential privacy: not all embedding dimensions carry equal privacy risk. Using differentiable mask learning, SPARSE identifies which dimensions encode specific privacy-sensitive concepts (names, locations, medical conditions), then applies calibrated noise — heavy on privacy-critical dimensions, minimal on semantically useful but non-sensitive ones.

For a topic matching system, this is particularly relevant: you could define "topic/domain" as the concept to preserve while adding noise to dimensions encoding authorship, project specifics, or organizational identity.

Status: Under peer review. Not yet widely validated beyond the authors' experiments.

Compass: Encrypted Semantic Search

Compass, published at OSDI 2025 by researchers at UC Berkeley, is the most impressive systems result in this space. It achieves accuracy matching plaintext search while ensuring data, query, and result privacy even if the server is fully compromised.

The system stores embeddings and HNSW graph indexes encrypted on the server, with search algorithms running locally on the client. Oblivious

RAM (ORAM) hides access patterns so the server cannot learn which embeddings are being compared. Innovations in directional neighbor filtering and speculative prefetching achieve user-perceived latencies around 1 second — dramatically faster than naive encrypted approaches.

Compass proves that encrypted semantic search at practical latencies is technically feasible. For a topic matcher, this means the matching infrastructure itself could be untrusted — neither the server operator nor an attacker who compromises it would learn anything about topics or queries.

Status: Academic paper at a top systems venue. Not yet productized. The engineering required to move from research prototype to production system is significant but not conceptually novel.

PIR-RAG: Query Privacy

PIR-RAG applies Private Information Retrieval to vector database queries. Documents are pre-clustered semantically; the client uses lattice-based homomorphic encryption to fetch cluster contents without the server learning which cluster was requested.

Recent advances in PIR-based vector search have shown promising results at large scale, with the field rapidly improving toward practical latencies. Combined with embedding encryption (Compass), PIR could provide both data-at-rest and query privacy — the server learns neither what's stored nor what's being searched for.

Status: Research prototype. Latency is improving rapidly but still above interactive thresholds for continuous matching.

MPC for Bilateral Matching

Secure Multi-Party Computation enables two parties to compute topic similarity without either revealing their embeddings. Using frameworks like Meta's CrypTen (GPU-accelerated MPC with PyTorch abstractions) or the ABY framework, both parties independently embed their topics, then use secret sharing to compute similarity scores. Each party learns only the final score — never the other's embedding or text.

Status: MPC frameworks are mature, with established use in sectors like financial services. Applying them to embedding similarity is straightforward engineering. The main constraint is computational overhead for high-volume, continuous matching.

The Honest Limitations

Even with layered defenses — both today's deployable stack and tomorrow's research prototypes — certain limitations are inherent to the problem.

The "small world" problem. In specialized domains, even abstract topic descriptions may be uniquely identifying. If only one team in a 10,000-person organization works on "post-quantum key exchange for embedded medical devices," no amount of embedding defense hides who registered that topic. Privacy guarantees are strongest in crowded topic spaces and weakest for genuinely novel work.

Composition attacks. Even if individual embeddings are well-protected, patterns of queries over time reveal organizational focus areas. Differential privacy composition theorems apply — the privacy budget depletes with each query. A system that runs continuously will eventually leak more than a snapshot would suggest.

Model update fragility. When embedding models are updated — as providers periodically release new versions — all stored embeddings must be regenerated. Eguard projections must be retrained. Encryption keys may need rotation. This operational burden is significant and often underestimated.

Adversarial insiders. Most defenses assume an honest-but-curious adversary — one that follows the protocol but tries to infer private information from what they observe. A malicious insider who controls the sanitization pipeline or the matching query mechanism can craft inputs specifically designed to probe for information about specific topics or individuals.

The sanitization-utility tension. Aggressive sanitization protects privacy but degrades match quality. Lenient sanitization preserves match quality but leaks context. Finding the right balance is domain-specific and requires ongoing tuning, not a one-time configuration.

The right framing is **defense-in-depth**, not perfect privacy. The goal is raising the cost of attack above the value of the information — the same principle behind encryption. No one claims AES prevents all data breaches; it makes them economically unviable for most adversaries. Embedding defenses should be evaluated the same way.

Practical Recommendations

If you store embeddings of sensitive data today

Audit your vector databases. Identify which collections contain embeddings of sensitive text (PII, medical records, financial data,

internal communications). Treat these with the same classification as the source data.

Deploy pre-embedding sanitization immediately. This is the highest-impact, lowest-effort defense. Use NER and PII detection to redact sensitive entities before embedding. Tools like Microsoft Presidio, SpaCy's NER, or commercial alternatives can be integrated into your embedding pipeline in days, not months.

Evaluate application-layer encryption. If your threat model includes database breaches or insider threats, encrypted embeddings (IronCore Labs' Cloaked AI or equivalent) prevent inversion attacks on stored data without changing your vector database infrastructure.

Do not rely on Gaussian noise alone. ZSinvert has demonstrated robustness against noise-based defenses. Noise may provide marginal protection but should not be your primary defense.

If you're planning cross-organizational intelligence

Start with the deployable architecture (sanitization + encryption + bilateral consent). Do not wait for research prototypes to mature.

Design the sanitization pipeline carefully. This is where most of the practical work lies. Topic descriptions should capture domain and problem type at a level abstract enough to find matches without revealing identifying details.

Plan for the "small world" problem. In organizations with highly specialized teams, consider whether topic matching adds enough value to justify the residual privacy risk. It works best in large, diverse organizations where many teams work on overlapping problems.

Watch the research. Eguard-style projections and Compass-style encrypted search are progressing toward production readiness. When they arrive, they can be layered into the architecture to significantly strengthen privacy guarantees.

What to tell your security team

Embeddings should be classified at the same sensitivity level as the source data they were generated from. A vector database containing embeddings of customer medical records is, for all practical purposes, a database containing customer medical records. Your DLP policies, access controls, encryption requirements, and retention policies should reflect this.

The good news is that defensible architectures exist today and are getting stronger. The bad news is that most organizations haven't deployed them yet — because they're still operating under the assumption that embeddings are safe.

That assumption is wrong. Now you know.

References

- Morris, J.X. et al. "Text Embeddings Reveal (Almost) As Much As Text." EMNLP 2023. arXiv:2310.06816
- Huang, Y.-H. et al. "Transferable Embedding Inversion Attack." ACL 2024. ACL Anthology
- Zhang et al. "ZSinvert: Universal Zero-Shot Embedding Inversion." 2025. arXiv:2504.00147
- Tonic.ai. "Sensitive Data in Text Embeddings Is Recoverable." 2024. tonic.ai/blog
- "Eguard: Defending LLM Embeddings Against Inversion Attacks." 2024. arXiv:2411.05034
- "SPARSE: Concept-Aware Privacy Mechanisms." Under review, ICLR 2026. OpenReview
- Zhu, J. et al. "Compass: Encrypted Semantic Search." OSDI 2025. USENIX
- "PIR-RAG: Private Information Retrieval for RAG." 2025. arXiv:2509.21325
- IronCore Labs. "Text Embedding Privacy Risks / Cloaked AI." 2024. ironcorelabs.com
- "CMAG: Metric Differential Privacy for Sentence Embeddings." ACM TOPS, 2025. ACM DL
- "Federated Semantic Learning for Cross-Domain Recommendation." ACM TOIS, 2025. ACM DL
- Perone, C.S. "Privacy-Preserving InferSent." 2018. blog
- Zhuang et al. "Rethinking the Privacy of Text Embeddings: A Reproducibility Study." 2025. arXiv:2507.07700