BeamClean: Language Aware Embedding Reconstruction

Kaan Kale
Protopia AI,
Austin, TX, USA
kaan.kale@protopia.ai

Kyle Mylonakis
Protopia AI,
Austin, TX, USA
kyle@protopia.ai

Jay Roberts
Protopia AI,
Austin, TX, USA
jay@protopia.ai

Sidhartha Roy
Protopia AI,
Austin, TX, USA
sid@protopia.ai

Abstract

In this work, we consider an inversion attack on the obfuscated input embeddings sent to a language model on a server, where the adversary has no access to the language model or the obfuscation mechanism and sees only the obfuscated embeddings along with the model's embedding table. We propose `BeamClean`, an inversion attack that jointly estimates the noise parameters and decodes token sequences by integrating a language-model prior. Against Laplacian and Gaussian obfuscation mechanisms, `BeamClean` always surpasses naive distance-based attacks. This work highlights the necessity for and robustness of more advanced learned, input-dependent methods.

1 Introduction

Machine learning services increasingly rely on shared or outsourced resources to manage and process data. Model-as-a-Service (MaaS) is a prime example, where organizations outsource the generation and storage of pre-trained large language models, computing infrastructure, and core functionalities that enable users to fine-tune, deploy, and execute customized models.

While this arrangement offers scalability and cost benefits, it also heightens privacy risks, particularly for Large Language Models (LLMs). Plaintext data must be transmitted from the client's trust zone to enter the model provider's. When the data is at rest in the client's trust zone or in transit to the model provider, the plaintext data may be protected via encryption, however, while the data is in use by an LLM, the data must be converted to either plaintext tokens or word embeddings (which are in 1-1 correspondence with the plaintext tokens). This need to expose data while it is in use presents a heightened privacy risk as this plaintext data could be leaked or intercepted.

To mitigate these privacy risks, recent approaches introduce input-independent noise to token embeddings in the local trust zone as in, Fig 1, before sharing with the service provider. One common method is local differential privacy techniques Mai et al. [2023], Shen et al. [2023], Feyisetan and Kasiviswanathan [2021], Carvalho et al. [2023] which add input independent noise to the embeddings. This paper investigates the possibility of inverting input-independent noise-perturbed input word embeddings back to their original token counterparts to recover the plaintext data.

We propose a novel attack strategy, `BeamClean` [1] (Fig 2), aimed at reconstructing sequences of tokens from their obfuscated embeddings. We assume a scenario, wherein the adversary has no access

[1] `https://github.com/beamclean-neurips25/beamclean`

Preprint. Under review.

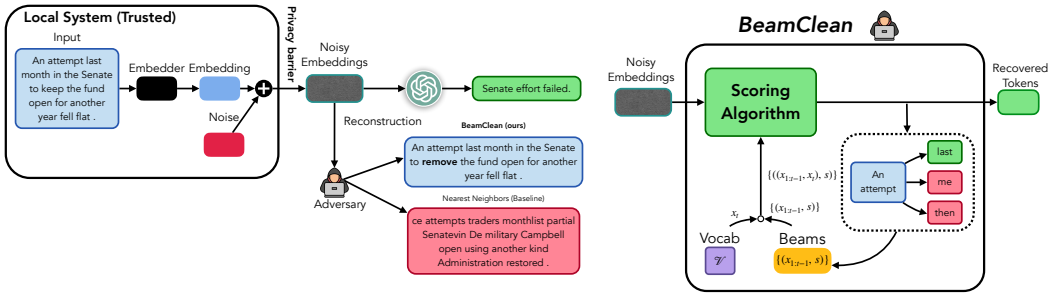


Figure 1: Overview of the generic input-embedding obfuscation pipeline and our adversarial threat model. Plaintext inputs are first encoded and transformed into noisy (i.e. obfuscated) embeddings, which are then transmitted to the LLM provider. An attacker accesses the noisy embeddings in order to attempt to recover the original plaintext input. Within the local trust zone, an obfuscation mechanism is applied to embeddings for a target LLM. These noisy embeddings are inputs to the `BeamClean` algorithm. The noisy embeddings are then put through a scoring algorithm to determine the top-k candidate token-ids. These top candidates are added to candidates from previously scored tokens to form beams. The top scoring beams are selected and used to start the scoring algorithm for the next token in the sequence.

to the target model's internal parameters, the obfuscation mechanism, or its training data. Instead, the attacker only observes the obfuscated embeddings Mai et al. [2023] and has knowledge of the underlying embedding table. This attack scenario could occur if an unauthorized user gains access to the server's compute infrastructure.

Our results demonstrate that `BeamClean` (Fig. 2) substantially outperforms nearest neighbor-based methods previously used to assess privacy strength Mai et al. [2023] against noise mechanisms that do not vary with input, effectively reconstructing plaintext sequences. This emphasizes a critical insight: Input independent Gaussian and Laplacian noise mechanisms, like those used in local-DP, can be vulnerable against adversaries equipped with linguistic priors and embedding knowledge.

The structure of this paper is as follows. Section 2 reviews the related work. Section 3 introduces the problem setting under consideration. In Section 4, we develop the framework and propose our attack method. Next, Section 5 presents experimental results to demonstrate the effectiveness of our proposed approach. Finally, Section 7 provides concluding remarks.

2 Related Work

With the growth of large language models (LLMs), chatbots such as ChatGPT, LLama Touvron et al. [2023], Achiam et al. [2023], and other embedding model services have raised significant privacy concerns. These services allow for sensitive or proprietary data to be transmitted during both training and inference, potentially leading to privacy risks. The main focus is to protect against these privacy risks without sacrificing model performance. Existing research primarily focuses on centralized learning and the leakage of training data in public LLM deployments, with attention given to pre-training Li et al. [2021], fine-tuning Yu et al. [2021], and prompt-tuning Li et al. [2023]. However, little work has addressed local privacyChatzikokolakis et al. [2013] during the inference phase.

Text Reconstruction from Contextualized Embeddings Previous studies have focused on inverting the contextualized embeddings produced by an *embedding model* Kugler et al. [2021], Morris et al. [2023]. These studies demonstrate that high BLEU scores can be achieved when an attacker accesses the model's output. In Morris et al. [2023], the authors introduced Gaussian noise as a simple obfuscation technique and presented preliminary results showing that, at a modest noise level, reconstruction accuracy dropped drastically (BLEU score fell from over 80 to around 10), while retrieval performance was barely affected. These findings suggest that noise injection could be a practical way to protect sensitive text data stored in vector databases without sacrificing search utility.

Input Data Protection in Split Learning In Mai et al. [2023], the authors propose a method for protecting input data in split learning by adding Laplacian noise under a local differential privacy framework. Their approach includes a pipeline that post-processes the contextualized embeddings generated by an embedding model. In Shen et al. [2023], a similar strategy is adopted with an additional step: after adding Laplacian noise, the noisy embedding is mapped to its nearest neighbor, resulting in a change of the corresponding input token.

Word Embedding Perturbation Mechanisms In Carvalho et al. [2023], the authors propose a truncated exponential mechanism that, rather than directly perturbing the continuous word embedding, assigns each candidate word a score based on its negative distance from the input, adds calibrated Gumbel noise to these scores, and selects the word with the highest noisy score as the privatized output. This method dynamically adapts the noise to the local density of the embedding space, ensuring that the selected token remains close to the original while providing formal privacy guarantees. In contrast, Xu et al. [2020] replaces standard spherical noise with elliptical noise sampled from a density proportional to $\exp\left(-\epsilon\|z\|_{\text{RM}}\right)$, where the regularized Mahalanobis norm adjusts the noise according to the covariance structure of the embedding space. The perturbed embedding is then mapped to its nearest token, thereby enhancing privacy in sparse regions without sacrificing overall utility. Meanwhile, Feyisetan et al. [2020] presents a mechanism under d_χ-privacy that maps each word into a high-dimensional embedding space, adds noise sampled from an n-dimensional distribution with density proportional to $\exp\left(-\epsilon\|z\|\right)$, and then maps the perturbed vector to its nearest neighbor in the vocabulary.

All three methods share the common step of adding input-independent noise to embeddings and then mapping back to the nearest token. Their key differences lie in the noise model and privacy framework: Carvalho et al. [2023] and Xu et al. [2020] operate under standard differential privacy Xu et al. [2020] adopt noise based on local geometry—whereas Feyisetan et al. [2020] adopts d_χ-privacy, which defines privacy with respect to a metric over the embedding space.

3 Problem Formulation

We consider a setting in which an attacker gains access to obfuscated embeddings derived from sensitive text, but lacks direct knowledge of how these embeddings were transformed. In this section, we formalize the obfuscation mechanism, define the blind attack scenario, and state the inversion problem underpinning our proposed attack strategy. Though the obfuscation mechanisms tested in this paper are input-independent, `BeamClean` is mathematically formulated for the input and sequence dependent cases.

We begin by outlining some useful notation to be used throughout the rest of the paper. Let $\mathcal{V} = \{w_1, \ldots, w_{|\mathcal{V}|}\}$ denote the vocabulary of the target language model, and let each vocabulary word $w_i \in \mathcal{V}$ have an associated clean embedding $x_i \in \mathbb{R}^d$ from the embedding table $\mathcal{X}$.

Given a token sequence $w_{1:T} = (w_1, \ldots, w_T)$ of length T, where each $w_t \in \mathcal{V}$ for $t \in [T] := \{1, \ldots, T\}$, we define $x_t := x(w_t)$ to be the embedding corresponding to token w_t. Thus, the sequence of embeddings is denoted by $x_{1:T} = (x_1, \ldots, x_T)$. Also, for notational simplicity, we use the expressions $p_\theta(x)$ and $p(x; \theta)$ interchangeably.

The *target language model* is the cloud-hosted LLM which processes obfuscated embeddings of users' plaintext data. An attacker obtains these obfuscated embeddings and has access only to that model's embedding table and vocabulary. An *obfuscation mechanism* is a map $\mathcal{O}(\cdot; \theta)$ between a clean embedding x_t to an *obfuscated* embedding y_t with $t \in [T]$, parameterized by θ: $y_{1:T} = \mathcal{O}\big(x_{1:T}; \theta\big) = \big(y_1, y_2, \ldots, y_T\big)$.

In this work, we focus on *additive-noise* mechanisms where $y_t = x_t + n_t, \forall t \in [T]$ and each noise term n_t is drawn from a distribution $p\big(n_t \mid x_{1:T}; \theta\big)$.

The noise is similar to the noise mechanisms used in differential privacy. We consider an **attack scenario**, in which the adversary has access only to: The *obfuscated embeddings* $y_{1:T}$ (leaked from some system). The *embedding table* $\mathcal{X}$ of the target language model, which maps each token $w_i \in \mathcal{V}$ to its clean embedding x_i. The attacker does not have the target model's internal parameters (i.e. the model weights or model architecture), nor knowledge of the obfuscation mechanism $\mathcal{O}(\cdot; \theta)$, nor its training data. Such a scenario may arise when only partial leaks reveal the initial embedding layer

and the obfuscated inputs, but no deeper components. This situation naturally arises when multiple LoRa finetuned adapters, which typically leave the embedding layer as it is, are associated with the same base model via inference engines such as vLLM Kwon et al. [2023].

We make the following novel contributions to the inversion of transformed token embeddings to text:

Generalized Attack Framework. We introduce a novel approach for attacks against additive-noise obfuscation methods on sequential data. Even without direct access to the obfuscation mechanism or its parameters, our method adapts to both input-independent and input-adaptive noise.

Improved Inversion over Nearest-Neighbor. For sequences of word embeddings, `BeamClean` outperforms nearest-neighbor based reconstruction attacks against input-independent noise obfuscations, similar to those used in local-DP Feyisetan and Kasiviswanathan [2021].

Language Aware Reconstruction. Our method performs denoising of token embedding sequences by modeling them as interdependent, rather than treating individual tokens in isolation. This approach enables the incorporation of sequential dependencies present in language as priors during reconstruction, leveraging a pretrained language model.

4 BeamClean

The core idea of `BeamClean` is to jointly estimate the noise model parameters and decode the original token sequence using a beam-search based procedure that leverages both the embedding table $\mathcal{X}$ of the target model and a language prior.

Let $p(y \mid x; \theta)$ denote the noise model—i.e., the probability of observing the noisy embedding y given its clean counterpart x. Our goal is to obtain a maximum-a-posteriori (MAP) estimate of the noise-model parameters θ conditioned on the observed sequence $y_{1:T}$. Applying Bayes' rule and marginalizing over the unknown clean token sequence $w_{1:T}$ yields the following objective:

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta \mid y_{1:T}) = \arg\max_{\theta} \log \sum_{w_{1:T} \in \mathcal{V}^T} \pi_\theta\big(y_{1:T} \mid x(w_{1:T})\big)\, p_{\mathrm{LM}}(w_{1:T}) + \log p(\theta) \tag{1}$$

In this formulation, because we assume an uninformative uniform prior, the $p(\theta)$ term is constant and drops out of the optimization. A full expression and detailed derivation are provided in Appendix A.2. The key terms in Equation (1) are as follows:

- π_θ, is the surrogate noise model. The likelihood term $\pi_\theta(y_{1:T} \mid x_{1:T})$ is factorized as: $\pi_\theta(y_{1:T} \mid x_{1:T}) \triangleq \prod_{t=1}^{T} \pi_\theta(y_t \mid x_{1:T})$. If there is more information available about the obfuscation mechanism it can be incorporated to the surrogate noise model by parametrizing it accordingly. For example if we know that the obfuscation mechanism is Gaussian we can model it as:
$$\pi_\theta(x(w_{1:t})) \;=\; \mathcal{N}\big(y_t \,\big|\, x_t + \mu_\theta(x_{1:t}),\, \Sigma_\theta(x_{1:t})\big), \tag{2}$$
where $\mu_{\hat{\theta}}(\cdot)$ and $\Sigma_{\hat{\theta}}(\cdot)$ are the mean and covariance predicted by the surrogate noise model, respectively, for each time step t.
- $p_{\mathrm{LM}}(w_{1:T})$ is the prior language model over the token sequence, defined as $p_{\mathrm{LM}}(w_{1:T}) = \prod_{t=1}^{T} p_{\mathrm{LM}}\big(w_t \mid w_{1:t-1}\big)$ and it assigns low probability to linguistically implausible sequences.

Once we have an estimate of the parameters $\hat{\theta}$, we can decode the token sequence by maximizing the posterior of the tokens $w_{1:T}$ given the observed transformed embedding $y_{1:T}$:

$$\hat{w}_{1:T} = \arg\max_{w_{1:T}} \log p(w_{1:T} \mid y_{1:T}; \hat{\theta}) = \arg\max_{w_{1:T}} \log \pi_{\hat{\theta}}\big(y_{1:T} \mid x(w_{1:T})\big)\, p_{\mathrm{LM}}(w_{1:T}) \tag{3}$$

Evaluating the sum in Equation (1) requires enumerating all $|\mathcal{V}|^T$ token sequences, which is prohibitive for realistic vocabulary sizes and sequence lengths. To achieve tractable computation, we rely on two complementary approximations:

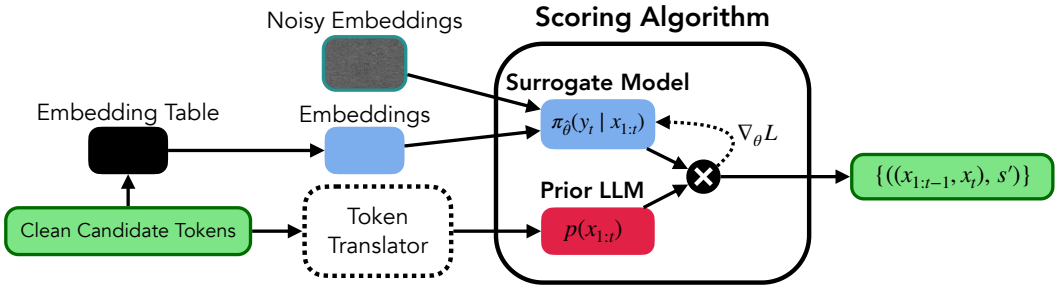


Figure 2: `BeamClean` is an iterative algorithm that begins with clean candidate tokens mapping to their corresponding embeddings. These clean candidate embeddings and noisy embeddings are inputs to a surrogate noise model of the obfuscation mechanism, $\pi_{\hat{\theta}}$. The clean candidate tokens are also used to produce a language prior (optionally translating tokens for the case of distinct target and prior language models). Together, the language prior and the surrogate noise model produce a likelihood score which is used to train the surrogate model. This is done iteratively to update the beam candidates. Finally, the highest scoring beam is selected as the reconstruction.

1. **Causal noise model.** We assume the obfuscation mechanism is causal,
$$p_\theta(y_t \mid w_{1:T}) \;=\; p_\theta(y_t \mid w_{1:t}), \qquad t < T,$$
so each noisy embedding depends only on the current and past clean tokens.
2. **Beam-search pruning.** Even with causality the number of candidate sequences still grows exponentially in T. We therefore keep only the top-B partial hypotheses at each time-step, selected by beam search (see Section 4).

Combining the causal assumption with beam-search pruning reduces the overall complexity from $\mathcal{O}(|\mathcal{V}|^T)$ to $\mathcal{O}(BT)$, making optimization feasible in practice.

Algorithm 1 Causal Beam Search Decoding with Adaptive Noise Estimation

Require: Vocabulary $\mathcal{X}$, beam size k, sequence length T, noisy embeddings $y_{1:T}$
1: Initialize beam $\mathcal{B}_0 \leftarrow \{(\texttt{""}, 1.0)\}$ ▷ Empty sequence with score 1
2: Initialize noise parameters $\theta^{(0)}$ (e.g. randomly or via a pre-training step)
3: **for** $t = 1$ **to** T **do**
4: $\quad\theta^{(t)} \leftarrow \arg\max_\theta \sum_{(x_{1:t-1},s)\in\mathcal{B}_{t-1}} \sum_{x_t\in\mathcal{X}} \pi_{\hat{\theta}}(y_t \,|\, x(w_{1:t}))\, p_{\mathrm{LM}}(w_t \mid w_{1:t-1})$
5: $\quad\mathcal{C} \leftarrow \varnothing$ ▷ Set of new candidates
6: $\quad$**for each** $(w_{1:t-1}, s)$ in $\mathcal{B}_{t-1}$ **do**
7: $\quad\quad$**for each** w_t in $\mathcal{X}$ **do**
8: $\quad\quad\quad s' \leftarrow s \;\times\; \pi_{\hat{\theta}}(y_t \mid x(w_{1:t})) \;\times\; p_{\mathrm{LM}}(w_t \mid w_{1:t-1})$
9: $\quad\quad\quad\mathcal{C} \leftarrow \mathcal{C} \cup \{(w_{1:t-1}, w_t),\; s'\}$
10: $\quad\quad$**end for**
11: $\quad$**end for**
12: $\quad\mathcal{B}_t \leftarrow$ Top-k entries of $\mathcal{C}$ by score
13: **end for**
14:
15: **return** The highest-scoring sequence in $\mathcal{B}_T$.

We employ a beam search approach that iteratively refines both the noise parameters θ and the decoded tokens. At each time step t, we keep a "beam", $\mathcal{B}_{t-1}$ of candidate partial sequences, each with an associated score. Lines 1–2 initialize an empty beam with the start-of-sequence hypothesis. At each time step t (lines 3–5), the surrogate noise parameters $\theta^{(t)}$ are updated by maximizing the joint likelihood of the current noisy embedding and all beam sequences. Lines 6–9 then extend each beam hypothesis by every token in the vocabulary, computing a new score by multiplying the surrogate likelihood (Eq.) with the language-model prior. Finally, lines 10–11 prune to the top-B sequences to form the next beam, and this process repeats until T, with the best scoring sequence returned.

Adaptive Noise Estimation. At each time step, re-estimating θ with the current beam $\mathcal{B}_{t-1}$ ensures that the model continually refines its approximation of the true obfuscation process. This incremental approach leverages context gained from earlier steps to improve parameter estimates.

Efficient Parameter Initialization. Using $\theta^{(t-1)}$ as the initialization for $\theta^{(t)}$ reflects the assumption that a single noise model operates across the entire sequence. Rather than refitting from scratch at each step, we exploit continuity in the underlying noise parameters.

Overall, this methodology interleaves adaptive noise-model estimation with a causal beam search for token decoding. By balancing linguistic plausibility (via a language prior) with consistency under the learned noise distribution, our approach substantially improves inversion accuracy over naive baselines, especially for input-independent noise mechanisms.

5 Experiments

We evaluate `BeamClean` across multiple datasets and Gaussian and Laplacian noise mechanisms, similar to those used in local-DP. We analyze performance under varying levels of adversarial knowledge and noise complexity. `BeamClean` is compared against the common baseline in blind-obfuscation literature of *Nearest Neighbor* Mai et al. [2023], Du et al. [2023], Xu et al. [2020], which decodes each noisy embedding y_t to the single clean embedding $x \in \mathcal{X}$ that minimizes $\|y_t - x\|_2$.

Our study uses three datasets. The first is constructed from randomly sampled examples in the Open-Orca dataset Mukherjee et al. [2023]. To standardize the input length, we truncate all sequences to a maximum of 32 tokens. The second dataset is the MRPC dataset from the GLUE benchmark Wang et al. [2019]. Finally, we use the PAPILLON dataset to evaluate PII recovery rate Siyan et al. [2025].

Following our problem formulation (Section 4), we incorporate a pretrained Llama-3.2-1B-Instruct model as the language prior in Equation (1).

We report the *attack success rate* (ASR), the fraction of correctly recovered tokens in a sequence:

$$\text{ASR} \;=\; \frac{\text{Number of Correctly Decoded Tokens}}{\text{Total Number of Tokens}} \times 100\%.$$

Higher values indicate more successful reconstructions.

For the PAPILLON benchmark, we report the mean PII recovery percent. For each sample in the dataset, we use PAPILLON to measure the percentage of PII strings leaked in the sample (based on their prompt [2]). We normalize PII recovery percent by the total number of PII strings detected in the clean sample and report the average value across the dataset.

To assess how our attacks perform under different privacy levels, we compute the corresponding differential privacy parameter ϵ for various noise magnitudes in the input-independent setting. The calculations for each noise distribution are provided in Appendix C.

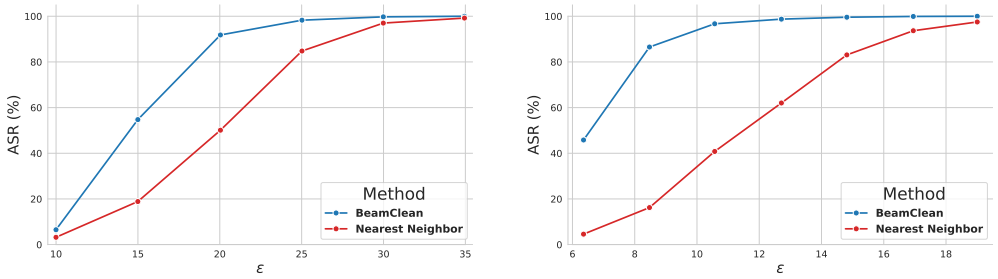
Our experiments were executed on NVIDIA H100 GPUs with 80 GB of memory. Training time grows approximately linearly with the beam width, dataset size, embedding-table size, and the number of candidate tokens considered.

5.1 Results

BeamClean always outperforms Nearest Neighbor. Figure 3 presents an experiment conducted on the MRPC dataset, where both the target model (including its embedding table) and the Prior Model are Llama-3.2-1B-Instruct with a vocabulary of size 128,256 Touvron et al. [2023]. The obfuscation is performed using isotropic Gaussian and Laplacian noise centered at zero, with the variance adjusted to different levels.

`BeamClean` consistently surpasses the nearest-neighbor baseline across all noise regimes. In particular, at stringent privacy settings (low ϵ, i.e. high noise), it delivers roughly 32% improvement in token-recovery rate—demonstrating its robustness and making it a clear choice when privacy guarantees tighten. Furthermore, we show that the variance can be learned during training, and our

[2] `https://github.com/Columbia-NLP-Lab/PAPILLON`



(a) Gaussian noise mechanism

(b) Laplace noise mechanism

Figure 3: Performance of `BeamClean` compared to Nearest Neighbor on the MRPC dataset. Curves show token-recovery rate as a function of ϵ with beam size 20. We compare against Gaussian, 3a, and Laplacian, 3b, noise mechanisms using, respectively. In both cases `BeamClean` outperforms Nearest Neighbor. Against Gaussian noise at $\epsilon = 15$ our attack recovers 74.3% of tokens versus 42.1% for Nearest Neighbor. Against Laplacian noise at $\epsilon = 8.5$ the attack attains 86% recovery versus 18% for Nearest Neighbor.

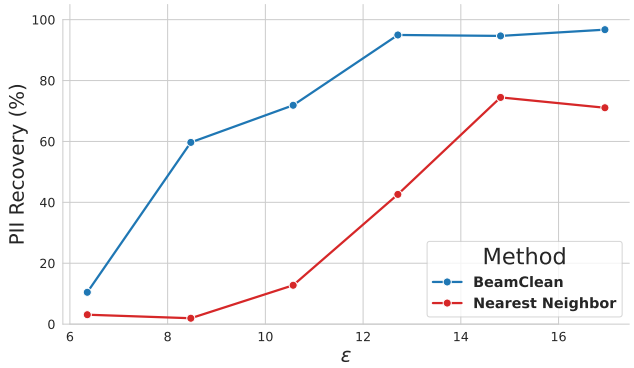


Figure 4: Mean PII Recovery percent on PAPILLON vs Laplace noise mechanism DP-ϵ. `BeamClean` consistently able to recover more PII strings than Nearest Neighbor, recovering 60.0% of PII strings compared to 1.9% recovered by Nearest Neighbor at $\epsilon = 8.5$.

decoding procedure compensates for stochasticity even at high variance. These findings are especially relevant given that prior work Mai et al. [2023] applies local differential privacy (LDP) techniques to both input and output embeddings, typically comparing the obfuscated embeddings against a nearest-neighbor attack to demonstrate privacy strength. In contrast to the nearest-neighbor attack, `BeamClean` recovers a substantially higher fraction of tokens. Notably, expanding the beam size or increasing the candidate pool can further improve performance, albeit with additional computational costs.

BeamClean recovers significantly higher PIIs compared to Nearest Neighbor. Figure 4 demonstrates the ability of `BeamClean` and Nearest Neighbor to recover PII strings from embeddings obfuscated with Laplacian noise mechanisms. For all the cases, `BeamClean` recovers more PII tokens than Nearest Neighbor. In particular, for an ϵ value of 8.5 Nearest Neighbor can only recover 1.9%, while `BeamClean` recovers 60% of PII strings. This underscores the importance of having stronger privacy attacks to measure the obfuscation quality. In this situation, a practitioner may believe that they had protected almost all their PII data when in actuality less than a third would have been protected from `BeamClean`.

The attacker does not need access to the target language model. We evaluate a scenario where the target embeddings originate from a GPT-2 embedding table, yet the decoding prior is provided by a Llama-3.2-1B-Instruct model. Because these two models use different tokenization schemes, each GPT-2 token must be mapped or approximated to a corresponding LLaMA token. We build a direct 1:1 mapping for each GPT-2 token to a unique LLaMA version of that token by choosing a restricted

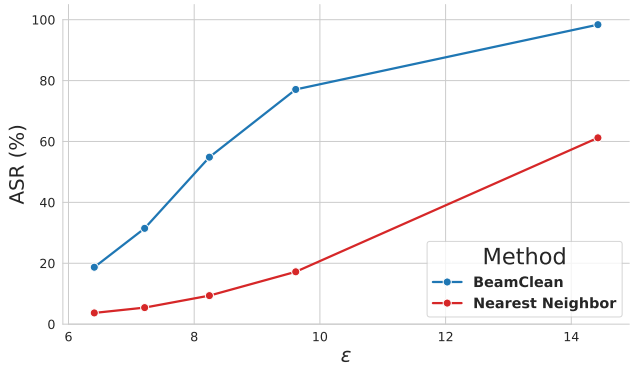


Figure 5: Attacking obfuscated GPT-2 embeddings using a Llama-3.2-1B-Instruct model as a language prior. `BeamClean` uniformly outperforms Nearest Neighbors, with the largest measured difference between the reconstruction methods occurring at DP $\epsilon \approx 9.6$, `BeamClean` achieves 77% recovery versus 17% for the baseline.

vocabulary, and we further restrict tokens to those present in MRPC. These experiments are run on the MRPC dataset from the GLUE benchmark.

Figure 5 shows the performance of `BeamClean` and Nearest Neighbor as a function of utility (ϵ). We see that even when the language prior is different from the target model, `BeamClean` dominates Nearest Neighbor, showing ASR nearly 60% higher compared to Nearest Neighbor when tested against Laplacian noise with a scale of 0.6.

6 Limitations

Although `BeamClean` demonstrates strong performance, several factors limit real-world applicability of this work. In our experimentation, we restricted ourselves to input independent noise mechanisms found in the literature. Though `BeamClean` is also applicable to more sophisticated input-dependent noise mechanisms, further experimentation is needed to determine its efficacy. From an algorithmic perspective, errors in decoding the earlier tokens in the sequences can propagate and escalate due to the autoregressive nature of language modeling. The beam strategy mitigates but does not eliminate, this risk. Further, large vocabularies and longer sequences greatly expand the search space, making exhaustive decoding expensive. Our beam-pruned approach helps, yet remains GPU-intensive. Despite these drawbacks, our results highlight that constant-noise mechanisms may be significantly more vulnerable to systematic attacks than prior work Mai et al. [2023] suggests.

7 Conclusion & Future Work

We presented a novel attack framework for inverting obfuscated embeddings under a scenario wherein the adversary can only access leaked, noise-perturbed embeddings and the target language model's embedding table. `BeamClean` combines learned noise-model estimation with language model priors in order to decode obfuscated embeddings to recover plaintext more effectively compared to naive nearest neighbor-based baselines used in input-independent Gaussian and Laplacian noise mechanisms. Moreover, we showed that `BeamClean` was able to consistently recover a significantly higher percentage of PII compared to the Nearest Neighbor attack. We also demonstrated that `BeamClean` does not necessarily need access to the target language model for the attack to be successful, inverting nearly 60% more tokens than Nearest Neighbor when the decoding prior is a different model than the target language model.

Future work could investigate how well `BeamClean` can reconstruct embeddings obfuscated with input *dependent* noise. Additionally, varying hyperparameters, noise distributions, and model architectures further influence both obfuscation strength and attacker capabilities. Likewise, incorporating domain constraints, such as partial vocabulary knowledge or specialized language priors, offers promising directions for refining `BeamClean` and improving privacy-preserving designs in MaaS environments.

8 Ethics Statement

Our work introduces a state-of-the-art inversion of noise-perturbed embeddings back to plaintext. This plaintext could include prompts from users that could contain sensitive and proprietary information and so `BeamClean` represents an additional avenue of privacy leakage not previously addressed in the literature. We believe that the description and release of `BeamClean`, along with the associated source code and data, will enable security researchers to have a stronger method to measure the protections offered by their privacy enhancing technologies.

References

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. Split-and-denoise: Protect large language model inference with local differential privacy. *arXiv preprint arXiv:2310.09130*, 2023.

Xicong Shen, Yang Liu, Huiqi Liu, Jue Hong, Bing Duan, Zirui Huang, Yunlong Mao, Ye Wu, and Di Wu. A split-and-privatize framework for large language model fine-tuning. *arXiv preprint arXiv:2312.15603*, 2023.

Oluwaseyi Feyisetan and Shiva Kasiviswanathan. Private release of text embedding vectors. In Yada Pruksachatkun, Anil Ramakrishna, Kai-Wei Chang, Satyapriya Krishna, Jwala Dhamala, Tanaya Guha, and Xiang Ren, editors, *Proceedings of the First Workshop on Trustworthy Natural Language Processing*, pages 15–27, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.trustnlp-1.3. URL `https://aclanthology.org/2021.trustnlp-1.3/`.

Ricardo Silva Carvalho, Theodore Vasiloudis, Oluwaseyi Feyisetan, and Ke Wang. Tem: High utility metric differential privacy on text. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 883–890. SIAM, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

Yansong Li, Zhixing Tan, and Yang Liu. Privacy-preserving prompt tuning for large language model services. *arXiv preprint arXiv:2305.06212*, 2023.

Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In *Intl. Symposium on Privacy Enhancing Technologies Symposium*, 2013.

Kai Kugler, Simon Münker, Johannes Höhmann, and Achim Rettinger. Invbert: Reconstructing text from contextualized word embeddings by inverting the bert pipeline. *arXiv preprint arXiv:2109.10104*, 2021.

John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. Text embeddings reveal (almost) as much as text, 2023. URL `https://arxiv.org/abs/2310.06816`.

Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. A differentially private text perturbation method using a regularized mahalanobis metric. *arXiv preprint arXiv:2010.11947*, 2020.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. In *Proceedings of the 13th international conference on web search and data mining*, pages 178–186, 2020.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Minxin Du, Xiang Yue, Sherman S. M. Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, page 2665–2679. ACM, November 2023. doi: 10.1145/3576915.3616592. URL http://dx.doi.org/10.1145/3576915.3616592.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*, 2023.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding, 2019. URL https://arxiv.org/abs/1804.07461.

Li Siyan, Vethavikashini Chithrra Raghuram, Omar Khattab, Julia Hirschberg, and Zhou Yu. Papillon: Privacy preservation from internet-based and local language model ensembles, 2025. URL https://arxiv.org/abs/2410.17127.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284. Springer, 2006.

Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 351–360, 2009.

Mangesh Gupte and Mukund Sundararajan. Universally optimal privacy mechanisms for minimax agents. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 135–146, 2010.

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

A Inverse Problem

A.1 Single Token

We start with a non-sequential (single-step) inverse problem where we observe:

$$y = x + n, \quad n \sim \mathcal{M}(x; \mu(x), \Sigma(x)),$$

and wish to infer $x \in \mathbb{R}^d$. We have a prior $p(x)$ (e.g., learned from some large dataset such as Alpaca).

Find parameters θ that relate noisy data y to clean embeddings x. A typical Bayesian formulation:

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta \mid y) \tag{4}$$

$$= \arg\max_{\theta} \left[\log \int p(y \mid x, \theta)\, p(x)\, dx + \log p(\theta) \right]. \tag{5}$$

A.2 Sequential

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta \mid \mathbf{y}_{1:T})$$

$$= \arg\max_{\theta} \left[\log \int p(\mathbf{y}_{1:T} \mid \mathbf{x}_{1:T}, \theta)\, p(\mathbf{x}_{1:T})\, d\mathbf{x}_{1:T} + \log p(\theta) \right].$$

B Inverse Problem (Token Sequence)

We consider a sequence of noisy word embeddings $\mathbf{y}_{1:T} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$, where T is the sequence length. The corresponding clean embeddings (unobserved) are denoted as $\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$.

The noise model assumes that the noisy embedding $\mathbf{y}_t$ at time t depends on all previous clean embeddings $\mathbf{x}_{1:t}$, parameterized by θ:

$$p(\mathbf{y}_t \mid \mathbf{x}_{1:t}, \theta).$$

The prior on the clean embeddings $\mathbf{x}_{1:T}$ is given by a pretrained language model:

$$p(\mathbf{x}_{1:T}) = p(\mathbf{x}_1) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{<t}).$$

The goal is to maximize the marginal likelihood of the noisy embeddings:

$$\log p(\mathbf{y}_{1:T} \mid \theta) = \log \int p(\mathbf{y}_{1:T}, \mathbf{x}_{1:T} \mid \theta)\, d\mathbf{x}_{1:T}.$$

C Additive-Noise Mechanisms for Differential Privacy

1 Global Sensitivity

For a real-valued query $f : \mathcal{D} \to \mathbb{R}$, the (global) ℓ_p-sensitivity is

$$\Delta_p f \;=\; \max_{\substack{x,y \in \mathcal{D} \\ \|x-y\|_0 \le 1}} \big\| f(x) - f(y) \big\|_p, \tag{6}$$

i.e. the greatest change in the output when one record is added or removed. Equations (7)–(9) calibrate noise in terms of $\Delta_1 f$ or $\Delta_2 f$.

2 Laplace Mechanism (ε-DP) Dwork et al. [2006]

$$\boxed{\mathcal{M}_{\text{Lap}}(x;\, f, \varepsilon) \;=\; f(x) \;+\; \text{Laplace}\big(0,\; b = \tfrac{\Delta_1 f}{\varepsilon}\big)} \tag{7}$$

Adding *i.i.d.* Laplace noise with scale $b = \Delta_1 f / \varepsilon$ guarantees ε-differential privacy since

$$\frac{\Pr[\mathcal{M}_{\text{Lap}}(x) = z]}{\Pr[\mathcal{M}_{\text{Lap}}(y) = z]} \;\le\; e^{\varepsilon}, \quad \forall\, z,\; x \sim y.$$

Discrete variant. Replacing continuous Laplace noise by the *geometric* (discrete Laplace) distribution yields the universally utility-maximizing geometric mechanism Ghosh et al. [2009], Gupte and Sundararajan [2010].

3 Gaussian Mechanism (ε, δ-DP) Dwork et al. [2014]

$$\boxed{\mathcal{M}_{\text{Gauss}}(x;\, f, \varepsilon, \delta) \;=\; f(x) + \mathcal{N}\Big(0,\; \sigma^2 = \frac{2\,\ln\big(1.25/\delta\big)\,(\Delta_2 f)^2}{\varepsilon^2}\Big)} \tag{8}$$

With $0 < \varepsilon < 1$ and $0 < \delta < 1$, the variance choice above ensures (ε, δ)-differential privacy by bounding the overlap between the two Gaussian output distributions corresponding to neighboring datasets.

4 Vector-Valued Queries ($d > 1$)

For $f : \mathcal{D} \to \mathbb{R}^d$ ($d \ge 2$), add independent noise per coordinate:

$$\boxed{\mathcal{M}^{d}_{\text{Gauss}}(x) \;=\; f(x) + (\eta_1, \ldots, \eta_d), \qquad \eta_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}\big(0, \sigma^2\big)} \tag{9}$$

where σ^2 is given by Equation (8) and the sensitivity $\Delta_2 f$ is computed in the ℓ_2 norm.