EDR: Data Reconstruction Attack Exploiting Word Embedding by Metaheuristic in Federated Learning of Language Models

Anonymous ACL submission

Abstract

With the rapid advancements in computational linguistics, machine learning-driven natural language processing (NLP) systems have become essential tools across various industries. These systems significantly enhance data processing efficiency, particularly in text classification tasks. Federated training frameworks present a promising solution for improving data protection. However, the exchange of information during parameter updates still carries the risk of sensitive data leakage. In this context, we identify potential information security threats to text classifiers operating within federated training frameworks and systematically analyze the relationship between model parameters and training data. Based on our analysis, we propose a novel gradient-based data reconstruction attack technique, which leverages knowledge from the embedding layer, referred to as the Embedding Data Reconstruction (EDR) attack. Our approach begins by identifying a set of tokens derived from the gradients. We then process these tokens and employ a metaheuristic integrated framework that combines Simulated Annealing (SA) and Tabu Search (TS). This framework assists us in finding the optimal sentence ordering while avoiding local optima. Finally, we fine-tune the model using the gradients obtained from the embedding layer. Our experimental results demonstrate substantial improvements across multiple datasets, with the most significant enhancement observed in bigrams, showing an average increase of approximately 45%.

1 Introduction

Amidst the backdrop of fragmented multi-party data, increasingly stringent privacy regulations, and a growing demand for cross-institutional collaboration, Federated Learning (FL) has emerged as an effective approach for leveraging distributed data sources while protecting data privacy. Unlike traditional centralized training approaches, FL enables each participant to maintain its dataset in a local environment, requiring only periodic submissions of model parameters or gradients to a central server for aggregation. This decentralized architecture fundamentally reduces privacy risks, as no raw data leaves the data holders' premises (Wu et al., 2023). A prominent example of successful FL implementation is Google's Gboard, where user typing data remains secured on individual smartphones while only model updates are transmitted to central servers for improvement of predictive capabilities (Hard et al., 2018; Yang et al., 2019).

Notwithstanding these benefits, recent studies have revealed critical vulnerabilities in FL systems, where adversaries can reconstruct or partially retrieve local training data through intercepting and reverse-engineering parameter or gradient updates (Liu et al., 2022; Balunovic et al., 2022; Gupta et al., 2022; Li et al., 2023). More concerning is the emergence of sophisticated attacks where malicious actors can manipulate the model to perform large-scale data reconstruction (Zhao et al., 2023; Boenisch et al., 2023). These attacks exploit subtle patterns within model updates to extract sensitive textual information, including personally identifiable information and proprietary content. This privacy vulnerability has become a critical challenge for both the Natural Language Processing and FL research communities, particularly in the context of fine-tuning pre-trained language models (Xie and Hong, 2021; Elmahdy et al., 2022; Zhang et al., 2022, 2023; Chen et al., 2023). In response to the aforementioned privacy vulnerabilities in FL systems, our study undertakes a multi-faceted exploration of gradient-based privacy attacks. Our main contributions are:

- We first dive into the intricate mechanisms underlying the reconstruction of training data, innovatively integrating metaheuristics SA and TS in a gradient-based data reconstruction attack to optimize the search for the best-reconstructed sentence.

- We are the first to utilize the gradient information of individual tokens in the embedding layer to determine the correctness of sentence ordering positions.

- Our proposed attack method, EDR, through implementation and experimental evaluation, has shown that it can reconstruct far more private text compared to previous approaches. This superiority is particularly prominent when the batch size is 1 and 2.

2 Related Work

The recovery of training data from gradients has emerged as a critical privacy concern in machine learning, particularly in federated learning systems. Initial research by Zhu et al. (Zhu et al., 2019) revealed fundamental vulnerabilities in gradient-based methods, demonstrating the feasibility of reconstructing private training data through gradient leakage. Following the work, Zhao et al. (Zhao et al., 2020) proposed the method which improved reconstruction quality by extracting ground-truth labels from gradients and empirically demonstrating its advantages.

The focus of gradient-based attacks has gradually shifted towards language models, with several significant developments. Deng et al. (Deng et al., 2021) proposed gradient attack algorithms specifically designed for Transformer-based language models, highlighting the urgent need for robust privacy protection mechanisms. Building on this foundation, Gupta et al. (Gupta et al., 2022) presented FILM, demonstrating successful text reconstruction from large batch sizes in FL settings, which uses GPT architecture and achieves high recovery rates on large batches but with comparatively lower accuracy.

Balunovic et al. (Balunovic et al., 2022) developed LAMP, an approach that leverages auxiliary language models with continuous and discrete optimization methods to guide reconstruction towards natural language text while avoiding local minima, though it still suffers from some local optima challenges.

In this work, we propose EDR, a BERT-based reconstruction method combined with a hybrid simulated annealing–tabu search framework. This design balances exploration and exploitation, improving reconstruction under small batch sizes and overcomes limitations in token selection and sentence ordering found in prior work.

Other methods include DAGER (Petrov et al., 2024), which infers token relationships by leveraging gradient correlations from client-side attention layers. While this approach sidesteps explicit alignment via a proxy mechanism, it assumes decoder-based architectures and requires cloud-grade GPUs (A100/L4). In contrast, our method avoids such architectural shortcuts and performs competitively on consumer-grade hardware (RTX 4090), even with longer sequences and smaller batch sizes. FET (Gao et al., 2025) uses two-phase optimization on embedding gradients but is outperformed by EDR in ROUGE-2 under small to medium batches due to our more balanced search.

Recent studies have further expanded the capabilities of reconstruction attacks. Morris et al. (Morris et al., 2023) achieved significant accuracy in text recovery through embedding inversion techniques, while He et al. (He et al., 2023) and Luo et al. (Luo et al., 2022) enhanced the efficiency of gradient-based reconstruction methods. Notably, Xu et al. (Xu et al., 2023) proposed the CGIR attack, demonstrating effective reconstruction without relying on strong model assumptions. Most recently, Wang et al. (Wang et al., 2025) introduced ILAMP, incorporating sequence beam search to enhance LAMP's performance in token order recovery.

3 Preliminary

3.1 Gradient-Based Attacks

A gradient leakage attack occurs when an attacker attempts to exploit the gradient updates $\nabla_{\boldsymbol{\theta}i} gi$ sent from the client to the server during FL to infer the client-owned private data (x_i, y_i), where $g_i = \nabla_{\theta_i}\mathcal{L}(x_i, y_i)$ denotes the gradient of loss function $\mathcal{L}$ computed on the private data. This is possible because gradients encode not only model update directions but also information about the underlying training data, presenting a potential privacy-leakage vector, and the more precise the gradient updates, the higher the risk of private data exposure. Attackers can reconstruct input features, labels, or even entire training samples from the shared gradients. In such attacks, it is assumed that the server is honest-but-curious, meaning it follows the federated training protocol as required while having the potential to try to extract sensitive information from the shared gradients, which is in line with many practical FL scenarios where the trust in the server is not absolute.

A common approach, adopted by Zhu et al. in

their work on "DLG" (Zhu et al., 2019) and Deng et al. concerning "TAG" (Deng et al., 2021), involves solving an optimization problem to reconstruct the private data. This problem is formulated as:

$$arg\ min\ \delta(\nabla_{\boldsymbol{\theta}_i} g_i^*, \nabla_{\boldsymbol{\theta}_i} g_i) \tag{1}$$

where δ represents the distance measure between gradients, $\nabla_{\boldsymbol{\theta}_i} g^*$ is the gradient computed from reconstructed data (x_i^*, y_i^*), and $\nabla_{\boldsymbol{\theta}_i} g_i$ is the gradient computed from real training data (x_i, y_i) with model parameters $\boldsymbol{\theta}_i$ at layer i. The attacker aims to minimize this distance to recover the private training data.

Common Distance Measure: Various distance measures δ have been proposed to quantify the similarity between gradients, each with unique characteristics and advantages:

L2 Distance: Zhu et al. (Zhu et al., 2019) employed the squared Euclidean norm, which penalizes larger differences in gradient magnitude, providing a smooth optimization surface.

L1 and L2 Combined Distance: Deng et al. (Deng et al., 2021) adopted a hybrid approach that combines L1 and L2 distances, leveraging the robustness of L1 against outliers and the smooth optimization benefits of L2. Consider an l-layer network, where the variable θ_i represents the parameters of layer i.

$$\mathcal{L}_{\cos}(\boldsymbol{x}) = 1 - \frac{1}{l}\sum_{i=1}^{l} \frac{\nabla_{\boldsymbol{\theta}_i} g_i^* \cdot \nabla_{\boldsymbol{\theta}_i} g_i}{\|\nabla_{\boldsymbol{\theta}_i} g_i^*\|_2 \|\nabla_{\boldsymbol{\theta}_i} g_i\|_2}. \tag{2}$$

Cosine Similarity: Geiping et al. (Geiping et al., 2020) and Balunović et al. (Balunovic et al., 2022) used cosine similarity to measure the angular difference between two gradients, emphasizing directional consistency while ignoring magnitude differences.

$$\begin{aligned}\mathcal{L}_{\text{tag}}(\boldsymbol{x}) = &\sum_{i=1}^{l} \|\nabla_{\boldsymbol{\theta}_i} g_i^* - \nabla_{\boldsymbol{\theta}_i} g_i\|_2 \\ &+\alpha_{\text{tag}} \|\nabla_{\boldsymbol{\theta}_i} g_i^* - \nabla_{\boldsymbol{\theta}_i} g_i\|_1 .\end{aligned} \tag{3}$$

3.2 Neighborhood Search

Neighborhood search, a fundamental optimization technique, operates on the principle of exploring the neighborhood of the current solution to find better ones (Sacramento et al., 2019). It starts with a feasible initial solution, which can be either randomly generated or derived from simple heuristic methods. Once determined, this initial solution becomes the current solution. During the search, the current solution is iteratively updated to approach the optimal solution.

In this study, we employ three neighborhood operations—**Swap**, **Insert**, and **Reverse**—to systematically explore the solution space. The **Swap** operation exchanges the positions of two selected words, **Insert** relocates a word to a new position, and **Reverse** inverts the order of a selected sequence of words. These transformations enable diverse local adjustments, effectively refining the solution by mitigating suboptimal arrangements and guiding the search toward improved reconstructions.

3.3 Simulated Annealing Algorithm

As a local search metaheuristic algorithm, the SA algorithm can be used to solve both discrete and continuous optimization problems. Its core concept involves introducing randomness during the search to avoid being trapped in local optima. Simultaneously, by gradually reducing the randomness of the search by controlling the temperature parameter, the algorithm eventually converges to the global optimum (Bertsimas and Tsitsiklis, 1993). Its key advantage lies in its simplicity. It can be rapidly implemented without prior acquaintance with the problem structure, making it suitable for tackling computationally complex problems in many practical applications.

The basic procedure of the SA algorithm is as follows. Initially, an initial solution is randomly generated, and an initial temperature T is set. Then, in each iteration, a new solution is generated through a certain method. Calculate the difference ΔE between the objective function values of the new solution and the current solution. If $\Delta E < 0$, the new solution is accepted as the current solution. If $\Delta E > 0$, the new solution is accepted with a certain probability, which is usually $P = \exp(-\Delta E/T)$. As the iteration progresses, the temperature T is gradually decreased, causing the algorithm to be more inclined to accept solutions with better objective function values in the later stages. When the temperature drops to a sufficiently low level or other stopping conditions are met, the algorithm halts, and the solution at this point is regarded as an approximate global optimum (Jiao et al., 2020).

3.4 Tabu Search

TS provides an effective approach to solving complex optimization problems. Its core idea is to avoid the algorithm getting trapped in local optimal solutions by introducing a memory mechanism, commonly known as the tabu list (Gendreau and Potvin, 2005). This list records the solutions that have been visited or specific search actions, and prohibits the revisit of these solutions or actions within a certain period, thus guiding the search towards more promising regions (Lai and Fu, 2019; Wang et al., 2019).

Mathematically, the TS algorithm can typically be described as follows. Let the objective function of the optimization problem be $f(x)$, where x is the solution vector. The algorithm starts from an initial solution x_0 and seeks better solutions through a series of neighborhood searches. The neighborhood search is usually defined as a set of new solutions obtained by making small changes to the current solution. In each iteration, the algorithm selects the best candidate solution from the neighborhood of the current solution. If this candidate solution is not in the tabu list or it satisfies certain aspiration criteria (e.g., its objective function value is much better than the current optimal solution), then this candidate solution is accepted as the new current solution. Meanwhile, some relevant information is updated in the tabu list to prevent the algorithm from revisiting the same solutions or actions in the short term.

3.5 Threat Model

In this context, we set up an experimental environment without any security issues, thus there is no external attacker, as shown in the left-hand part of Figure 1. However, we designate the server as the attacker. The server, in this case, is both honest and curious, monitoring the communication between itself and a random client during the federated training of a language model, as previously elaborated. This server, masquerading as the aggressor, obtains white-box admittance to two essential parcels of information: 1) the gradients transmitted by the client and 2) the model parameters, including the vocabulary and the embedding matrix. It should be emphasized that the server, as the attacker, can inspect this information at any stage of the training process. The opponent's objective is to retrieve a minimum of one sentence from the set of confidential training data by exploiting the information available to him. This is crucial because obtaining even a single sentence is enough to compromise the privacy guarantees provided by FL. Furthermore, the adversary can repeat the attack on a single batch multiple times to extract more sentences. The extent of resemblance between the retrieved sentence and the original private sentence from the batch gauges the effectiveness of the attack.

4 Approach

We propose a novel method for reconstructing training data, which is divided into three innovative steps: Hierarchical Subword Assembly, Gradient-Guided SA-TS Optimization, and Gradient Analysis and Token Adjustment, as depicted in the right-hand part of Figure 1.

4.1 TokenFusion: Hierarchical Subword Assembly

We implement the strategy used by Gupta et al. (Gupta et al., 2022) and employ a gradient analysis technique to extract a set of tokens $T = \{t_1, t_2, \ldots, t_n\}$ from the token embedding gradients $\nabla f(w_i)$. Here, w_i denotes the embedding vector of the i-th token, and $f(\cdot)$ represents the model or loss function under consideration. Concretely, we examine $\nabla f(w_i)$ for non-zero rows, each corresponding to a particular token embedding that influences the gradient. By identifying these non-zero rows, we recover the set of tokens present in the batch without directly accessing the original text. Once the complete set of tokens T is obtained, we proceed with further processing or reconstruction tasks. Distinctively, after acquiring the comprehensive set of tokens T, we pioneer a novel approach to processing these tokens. We initiate a series of operations on the sub-words prefixed with "##" and their corresponding root words. Firstly, we eliminate the special symbols and preserve the valid tokens as root words. Subsequently, we explore diverse combinations of root words and sub-words while meticulously examining their lengths and spellings.

After initially encoding the qualified lexemes, we re-confirm whether these encodings fall within the predefined set. Only those lexemes and their associated encodings that meet all the stipulated criteria are retained. Through this process, incomplete sub-words are assembled into units with complete semantic meanings, and the proper conjugation of sub-words with root words is ensured. Conse-

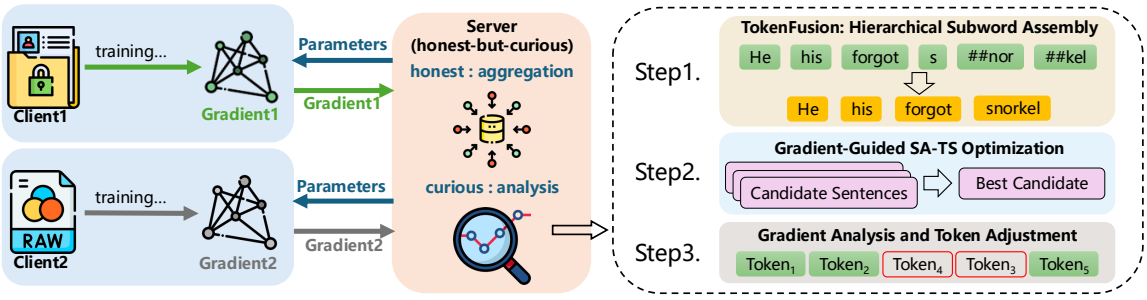

Figure 1: We assume that the attacker is an honest-but-curious server. The attack flow of EDR consists of three main steps: (i) Hierarchical Subword Assembly: The received gradient information is used to combine subwords (e.g., "##nor", "##kel") into a meaningful lexeme ("snorkel"). (ii) Gradient-Guided SA-TS Optimization: Candidate sentences are generated from candidate lexemes and refined using gradient information and leverage a hybrid approach combining SA and TS to select the best candidate sentence. (iii) Gradient Analysis and Token Adjustment: Token gradients of the best candidate are analyzed and iteratively adjusted to ensure precise alignment with target gradients, resulting in the reconstructed sentence that best matches the original data.

quently, in this step, a set of candidate lexemes $L = \{l_1, l_2, \ldots, l_m\}$ is obtained, where m may or may not be equal to n, the number of elements in the set of tokens $T = \{t_1, t_2, \ldots, t_n\}$. The set L is defined as:

$$L = \{\ell \mid \ell = Merge(t_1, t_2, \ldots, t_i), t_i \in T\} \quad (4)$$

which means that each element ℓ in L is formed by merging one or more tokens from the set T through the operation $Merge$. Each candidate lexeme in the set L may consist of a single token or multiple tokens, and every one of these candidate lexemes has the potential to appear in the training sentences we aim to reconstruct.

4.2 Gradient-Guided SA-TS Optimization

Unlike the method proposed by Gupta et al. (Gupta et al., 2022), which uses beam search for permutation, we employ a hybrid optimization framework integrating SA and TS for sentence configuration. The core objective is to identify the reconstructed sentence that minimizes the gradient distance. Importantly, the effectiveness of reconstruction is highly sensitive to batch size. Larger batches ($B = 4$) dilute gradient signals, hinder individual update extraction, and expand the combinatorial search space, increasing complexity and reducing optimization efficiency (Yue et al., 2023; Li et al., 2020; Wang et al., 2023). Additionally, large-batch normalization exacerbates non-IID effects, destabilizing model behavior. These issues highlight the advantage of small-batch settings, which preserve clearer gradients, motivating our focus on such configurations for reconstruction attacks.

Specifically, we first randomly sample from the candidate lexemes L to form multiple fixed-length sentences, constituting the initial population S. Then, within this hybrid framework, each sentence in S is reordered by exploring different permutations, and its gradient is compared against the target gradient. By iteratively refining sentence configurations to minimize the gradient distance, we ultimately obtain a reconstructed sentence that best aligns with the original data's gradient signals.

Evaluation function: SA requires an evaluation function to measure the quality of reconstructed sequences. Drawing on the findings of prior research, we adapt our evaluation function according to the batch size. When the batch size is 1, **Cosine Similarity** (Eq. 2) is adopted due to its high effectiveness in evaluating reconstructed sentences. For larger batch sizes, the **L1 and L2 Combined Distance** (Eq. 3) is employed to measure the similarity between the gradients of the generated and target sequences. The sequence with the lowest aggregated score or the highest cosine similarity is determined as the optimal one. This approach ensures that the evaluation criteria are in line with batch-specific conditions, and simultaneously quantifies the proximity to the original data through gradient similarity.

The hybrid SA-TS framework enhances optimization by combining SA's global exploration through stochastic acceptance with TS's memory-based avoidance of cycling, effectively preventing premature convergence and improving search efficiency in complex permutation spaces.

Through this integration (Fig. 2), the randomness

Algorithm 1 Initial population is optimized in the hybrid framework.

Input: Initial population S, Evaluation function $\mathcal{L}(S)$, Initial temperature T, Cooling rate α, Maximum iterations max_iter, Tabu list L
Output: Best sentence s^*
for s in S **do**
  Initialize $L \leftarrow L \cup \{s\}, failed \leftarrow 0$
  for $iter \in \{1, \ldots, max\_iter\}$ **do**
    Generate s' by **random shuffling**
    or **neighborhood operations**
    if $\mathcal{L}(s') - \mathcal{L}(s) < 0$ **then**
      $s \leftarrow s'$
    else
      $s \leftarrow s'$ with probability P
    end if
    if $s = s'$ **then**
      $L \leftarrow L \cup \{s'\}$
    else
      $failed \leftarrow failed + 1$
      if $failed > 0.1 \cdot max\_iter$ **then**
        break
      end if
    end if
    Reduce the temperature: $T \leftarrow \alpha \cdot T$
  end for
  if $\mathcal{L}(s) - \mathcal{L}(s^*) < 0$ **then**
    $s^* \leftarrow s$
  end if
end for
return s^*

and global searching power of SA are retained, allowing the search to jump out of local minima when needed, while TS systematically guides the search towards unexplored regions. As a result, the combined method both broadens and accelerates the overall optimization process, improving the likelihood of converging to a near-optimal or even globally optimal sentence arrangement. Through the implementation of this approach within the framework and the utilization of the evaluation function, we are able to obtain an optimal candidate sentence at this step, which has been optimized to minimize the gradient distance.

4.3 Gradient Analysis and Token Adjustment

In the final phase of our procedure, we conduct a detailed analysis of the embedding layer gradients corresponding to each token within the optimal candidate sentence obtained through the previous step. Our experiments reveal that when the target data has a batch size greater than 1, successfully restoring a single data entry causes the gradients of its tokens in the embedding layer to align with the target gradients. Consequently, only the remaining data entries require further adjustment.

To optimize the reconstructed sentences, we systematically evaluate the tokens within each sentence to ensure their positions are correct. Tokens identified as misaligned are flagged and iteratively adjusted. Same as the previous step, when the batch size is 1, cosine similarity (Eq. 2) is adopted. For larger batch sizes, the combined L1 and L2 combined distances (Eq. 3) are used as the benchmark to evaluate the alignment with the target gradients. Adjustments continue until the evaluation metrics reach their optimal values, ensuring precise alignment. The final output sentence represents the reconstructed sequence that achieves the highest similarity to the original data in terms of gradient alignment.

5 Experiments

5.1 Set Up

In our evaluation, we employ three pivotal binary text classification datasets to ensure a comprehensive analysis. Specifically, we utilize CoLA and SST-2 from the GLUE benchmark, along with the RottenTomatoes dataset—each featuring distinct sequence lengths. Our experiments are centered on the BERT$_{base}$ architecture provided by Hugging Face. We utilize the ROUGE metric suite, an approach also adopted in TAG (Deng et al., 2021) and LAMP (Balunovic et al., 2022). We calculate the aggregated F-scores for ROUGE-1, ROUGE-2, and ROUGE-L. ROUGE-1 is used to measure the accuracy of the recovered unigrams, ROUGE-2 measures the accuracy of the recovered bigrams, and ROUGE-L measures the ratio of the length of the longest matching subsequence to the length of the full sequence. Furthermore, when dealing with batches consisting of multiple sequences, we intentionally exclude the padding tokens from both the reconstruction process and the subsequent ROUGE computations. These padding tokens are used to standardize the lengths of sequences. By doing so, we ensure that our evaluation of the attack performance is accurate and is not affected by the artifacts introduced by padding.

Our method is compared with three main baselines, namely TAG, LAMP$_{\cos}$, and LAMP$_{\text{tag}}$,

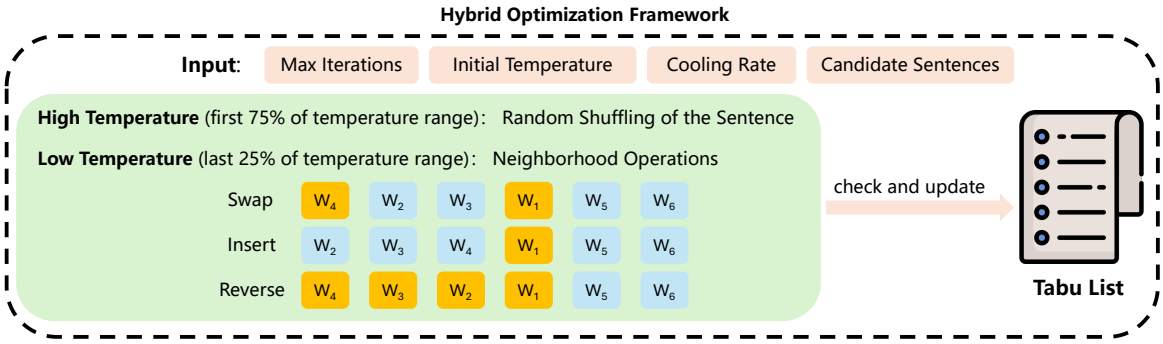

Figure 2: Gradient-guided Sequence Optimization Framework Using SA and TS

among which the two methods of LAMP are regarded as the current state-of-the-art methods. We use the open-source LAMP framework to implement it. To ensure that our method is compared to these baselines under fair conditions, all methods use prior knowledge.

In the configuration of our method, batch size directly influences key hyperparameters such as initial temperature, cooling rate, and iteration count, allowing us to balance global exploration and local refinement. We divide the temperature schedule into two phases: a high-temperature stage for broad search and a low-temperature stage for fine-tuning. All experiments are conducted on a workstation with an Intel Core i9-14900K CPU, 64GB RAM, and an NVIDIA RTX 4090 GPU (24GB VRAM).

Several key parameters must be predefined. These include the initial temperature, cooling rate, and maximum number of iterations. The maximum number of failed attempts is set at 0.1 of the maximum number of iterations. When considering different batch sizes, specific parameter settings are as follows. For a batch size of 1, the initial temperature is set at 300, and the cooling rate is 0.95. The maximum number of iterations is 3,000. In the case of a batch size of 2, the initial temperature is increased to 400, the cooling rate is adjusted to 0.99, and the maximum number of iterations reaches 4,000. For a batch size of 4, the initial temperature is further elevated to 500, the cooling rate remains at 0.99, and the maximum number of iterations is set at 5,000. These settings are carefully calibrated to optimize the performance of our method under various batch size conditions.

The above parameter choices were determined through systematic grid search experiments, ensuring a balanced trade-off between reconstruction quality and computational efficiency. Detailed tuning procedures and selection rationale are provided in Appendix B

5.2 Results and Analysis

As shown in Table 1, our method consistently outperforms baseline approaches across all datasets and evaluation metrics (R-1, R-2, and R-L). On CoLA, our method achieves remarkable results, with R-1, R-2, and R-L scores of 98.52, 93.42, and 96.01, respectively, for B=1, and maintains its lead for B=2 and B=4 with the highest R-L scores of 84.86 and 56.55. Similarly, on SST-2, the EDR approach achieves the best performance across all batch sizes, including a standout R-L score of 91.17 for B=1 and 65.64 for B=4. These results highlight its robustness in both small and large-batch scenarios. For the Rotten Tomatoes dataset, our method demonstrates superior performance in most metrics. At B=1, it achieves R-1, R-2, and R-L scores of 89.76, 29.98, and 57.64, significantly surpassing all baselines. Although the R-2 score for B=4 (4.24) is slightly lower than the best baseline, it remains comparable, while R-1 and R-L still lead at 44.33 and 29.05, respectively. Across all datasets and evaluation conditions, our approach excels by effectively balancing exploration and exploitation, achieving both global diversity and local precision in sentence reconstruction.

Several observations emerge from Table 2. First, our method consistently produces reconstructions that are significantly closer to the reference sentences across all datasets compared to $\text{LAMP}_{\cos}$. On the CoLA dataset, our method perfectly reconstructs the sentence, retaining its semantic and syntactic accuracy, while $\text{LAMP}_{\cos}$ introduces lexical errors such as "mykel snor," distorting the original meaning.

For SST-2, our method demonstrates superior capability in preserving both structure and meaning, closely matching the reference. In contrast,

		B=1			B=2			B=4		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
CoLA	TAG	84.61	12.12	55.59	78.00	12.35	53.76	65.22	8.20	48.79
	$\text{LAMP}_{\cos}$	88.54	52.28	75.26	77.56	32.14	64.51	61.99	18.09	53.01
	$\text{LAMP}_{\text{L1+L2}}$	87.70	48.06	73.96	81.13	35.89	66.56	68.48	20.03	55.15
	EDR	**98.52**	**93.42**	**96.01**	**95.54**	**74.30**	**84.86**	**68.51**	**25.00**	**56.55**
SST-2	TAG	79.96	18.50	58.94	76.61	17.62	57.08	66.39	13.20	51.41
	$\text{LAMP}_{\cos}$	88.05	58.22	77.50	79.43	41.82	69.03	63.73	26.34	56.59
	$\text{LAMP}_{\text{L1+L2}}$	89.57	62.44	77.31	85.64	48.33	72.96	74.82	33.56	63.20
	EDR	**97.34**	**84.46**	**91.17**	**89.92**	**65.99**	**78.66**	**74.96**	**43.43**	**65.64**
Rotten Tomatoes	TAG	68.76	3.40	35.68	54.52	3.02	32.46	43.98	1.97	29.12
	$\text{LAMP}_{\cos}$	65.61	10.49	39.80	53.80	9.62	37.10	38.54	3.14	28.03
	$\text{LAMP}_{\text{L1+L2}}$	70.09	5.85	34.94	58.25	8.66	36.62	41.74	**4.85**	28.96
	EDR	**89.76**	**29.98**	**57.64**	**69.45**	**16.62**	**42.17**	**44.33**	4.24	**29.05**

Table 1: EROUGE Score Comparison for Reconstruction Methods Across Datasets (Batch Sizes = 1, 2, and 4).

		Sequence
CoLA	**Reference**	**Who has seen my snorkel?**
	$\text{LAMP}_{\cos}$	**who has mykel snor seen?**
	EDR	**Who has seen my snorkel?**
SST-2	**Reference**	**ably balances real - time rhythms with propulsive incident.**
	$\text{LAMP}_{\cos}$	**ab balances - real time propulsively rhythms with incident.**
	EDR	**ably balances real - time rhythms with propulsive incident.**
Rotten Tomatoes	**Reference**	**vaguely interesting, but it's just too too much .**
	$\text{LAMP}_{\cos}$	**but just s too vaguely vaguely, just vaguely much much.**
	EDR	**vaguely interesting, but it's just too too much .**

Table 2: Comparison of Sentence Reconstruction Between EDR and $\text{LAMP}_{\cos}$ on Batch Size = 1

$\text{LAMP}_{\cos}$ fails to maintain grammatical coherence, introducing errors such as "ab balances" and "propulsively," which diverge from the intended semantics.

Similarly, on the Rotten Tomatoes dataset, our method successfully captures the tone and lexical nuances of the sentence with high fidelity. $\text{LAMP}_{\cos}$, however, generates a disjointed output with repeated and misplaced words, disrupting the fluency and interpretability.

These qualitative results demonstrate the effectiveness of EDR in reconstructing sentences with superior grammatical, lexical, and semantic alignment, particularly in small-batch settings, where precision is critical. EDR also offers strong computational efficiency and scalability, with favorable time complexity across batch sizes and sequence lengths, as elaborated in the Appendix A.

6 Conclusion

We propose a gradient-based data reconstruction technique. Our approach makes use of traditional gradient-reconstruction methods and prior knowledge and incorporates meta-heuristic algorithms to assist in the reconstruction of training data. This technique enhances the reconstruction accuracy for different datasets and batch sizes.

7 Limitations

From the experimental data, it can be seen that the main limitation of our attack method lies in reconstructing longer sentences. In particular, the sentence lengths in the Rotten Tomatoes dataset usu-

ally range from 14 to 27. This poses a huge search-space challenge for the ranking process. When the batch size is small, the success rate can be increased by 5% to 20%. However, when the batch size is 4, the performance is slightly better than that of the baseline. This result indicates that when the length of sentences is fixed in a proper range, our method is still highly applicable. In general, the proposed EDR significantly improves the attack rate in the context of data reconstruction for federated learning of the language model. When the batch size increases, the effectiveness of our method does not scale up as expected, which may be due to the increased complexity of handling multiple sentences simultaneously and the potential interference between them. This suggests that further optimization is needed to improve the performance of our method in scenarios with larger batch sizes and longer sentences.

8 Ethical Considerations

This research focuses on data reconstruction attacks in federated learning systems, aiming to highlight potential privacy vulnerabilities. While these findings are intended to improve the security and robustness of machine learning frameworks, we acknowledge the ethical implications associated with reconstructing private training data. The techniques developed could be misused if applied maliciously to compromise users' sensitive information.

To mitigate such risks, we emphasize that our work serves as a cautionary study to inform the design of more secure federated learning protocols and inspire the development of effective defense mechanisms. Researchers and practitioners should apply these methods responsibly, adhering to legal and ethical guidelines, and ensuring that privacy protections are strengthened rather than weakened. We advocate for transparency, informed consent, and strict data governance in any practical deployment involving user data.

References

Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev. 2022. Lamp: Extracting text from gradients with language model priors. *Advances in Neural Information Processing Systems*, 35:7641–7654.

Dimitris Bertsimas and John Tsitsiklis. 1993. Simulated annealing. *Statistical science*, 8(1):10–15.

Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. 2023. When the curious abandon honesty: Federated learning is not private. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 175–199. IEEE.

Chaochao Chen, Xiaohua Feng, Jun Zhou, Jianwei Yin, and Xiaolin Zheng. 2023. Federated large language model: A position paper. *arXiv preprint arXiv:2307.08925*.

Jieren Deng, Yijue Wang, Ji Li, Chenghong Wang, Chao Shang, Hang Liu, Sanguthevar Rajasekaran, and Caiwen Ding. 2021. TAG: Gradient attack on transformer-based language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3600–3610.

Adel Elmahdy, Huseyin A Inan, and Robert Sim. 2022. Privacy leakage in text classification a data extraction approach. In *Proceedings of the Fourth Workshop on Privacy in Natural Language Processing*, pages 13–20.

Ying Gao, Yuxin Xie, Huanghao Deng, and Zukun Zhu. 2025. Gradient inversion attack in federated learning: Exposing text data through discrete optimization. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2582–2591.

Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in neural information processing systems*, 33:16937–16947.

Michel Gendreau and Jean-Yves Potvin. 2005. Tabu search. *Search methodologies: introductory tutorials in optimization and decision support techniques*, pages 165–186.

Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. 2022. Recovering private text in federated learning of language models. *Advances in neural information processing systems*, 35:8130–8143.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*.

Xing He, Changgen Peng, Weijie Tan, et al. 2023. Fast and accurate deep leakage from gradients based on wasserstein distance. *International Journal of Intelligent Systems*, 2023.

Shanshan Jiao, Zhisong Pan, Yutian Chen, and Yunbo Li. 2020. Cloud annealing: a novel simulated annealing algorithm based on cloud model. *IEICE TRANSACTIONS on Information and Systems*, 103(1):85–92.

Xiangjing Lai and Zhang-Hua Fu. 2019. A tabu search approach with dynamical neighborhood size for solving the maximum min-sum dispersion problem. *IEEE Access*, 7:181357–181368.

Jianwei Li, Sheng Liu, and Qi Lei. 2023. Beyond gradient and priors in privacy attacks: Leveraging pooler layer inputs of language models in federated learning. In *International Workshop on Federated Learning in the Age of Foundation Models in Conjunction with NeurIPS 2023*.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60.

Pengrui Liu, Xiangrui Xu, and Wei Wang. 2022. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. *Cybersecurity*, 5(1):4.

Zeren Luo, Chuangwei Zhu, Lujie Fang, Guang Kou, Ruitao Hou, and Xianmin Wang. 2022. An effective and practical gradient inversion attack. *International Journal of Intelligent Systems*, 37(11):9373–9389.

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

Ivo Petrov, Dimitar I Dimitrov, Maximilian Baader, Mark Müller, and Martin Vechev. 2024. Dager: Exact gradient inversion for large language models. *Advances in Neural Information Processing Systems*, 37:87801–87830.

David Sacramento, David Pisinger, and Stefan Ropke. 2019. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C: Emerging Technologies*, 102:289–315.

Bin Wang, Qiang Zhang, and Xiaopeng Wei. 2019. Tabu variable neighborhood search for designing dna barcodes. *IEEE Transactions on NanoBioscience*, 19(1):127–131.

Jiajie Wang, Shuai Zhang, Yueling Xu, Zijian Zhang, Xuan Yang, and Yuanzhe Cheng. 2025. Ilamp: Improved text extraction from gradients in federated learning using language model priors and sequence beam search. *Expert Systems with Applications*, page 126592.

Yanmeng Wang, Qingjiang Shi, and Tsung-Hui Chang. 2023. Why batch normalization damage federated learning on non-iid data? *IEEE transactions on neural networks and learning systems*.

Feijie Wu, Zitao Li, Yaliang Li, Bolin Ding, and Jing Gao. 2023. Fedbiot: a solution for federated large language model fine-tuning with intellectual property protection.

Shangyu Xie and Yuan Hong. 2021. Reconstruction attack on instance encoding for language understanding. In *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP'21)*, pages 2038–2044.

Xiangrui Xu, Pengrui Liu, Wei Wang, Hong-Liang Ma, Bin Wang, Zhen Han, and Yufei Han. 2023. Cgir: Conditional generative instance reconstruction attacks against federated learning. *IEEE Transactions on Dependable and Secure Computing*, 20(6):4551–4563.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19.

Kai Yue, Richeng Jin, Chau-Wai Wong, Dror Baron, and Huaiyu Dai. 2023. Gradient obfuscation gives a false sense of security in federated learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 6381–6398.

Zhuo Zhang, Yuanhang Yang, Yong Dai, Lizhen Qu, and Zenglin Xu. 2022. When federated learning meets pre-trained language models' parameter-efficient tuning methods. *arXiv preprint arXiv:2212.10025*.

Zhuo Zhang, Yuanhang Yang, Yong Dai, Qifan Wang, Yue Yu, Lizhen Qu, and Zenglin Xu. 2023. Fedpetuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models. In *Annual Meeting of the Association of Computational Linguistics 2023*, pages 9963–9977. Association for Computational Linguistics (ACL).

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. 2020. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*.

Joshua Christian Zhao, Atul Sharma, Ahmed Roushdy Elkordy, Yahya H Ezzeldin, Salman Avestimehr, and Saurabh Bagchi. 2023. Loki: Large-scale data reconstruction attack against federated learning through model manipulation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 30–30. IEEE Computer Society.

Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep leakage from gradients. *Advances in neural information processing systems*, 32:1–11.

A Time Complexity Analysis of EDR

We provide a detailed analysis of the computational complexity of EDR, structured by its three main phases but presented compactly.

Hierarchical Subword Assembly (TokenFusion). This phase extracts token candidates from embedding gradients and combines them into valid lexemes. The gradient analysis costs $O(V \cdot d)$, where V is the vocabulary size and d is the embedding

dimension. Subword combination (e.g., handling "##" tokens) takes $O(k^2)$, where k is the number of subwords per candidate. Lexeme validation costs $O(m)$ for m candidates. Since $m \ll V$, this phase is dominated by $O(V \cdot d)$, which we approximate as $O(V + B \cdot P)$ in practice, where B is batch size and P is sentence length.

Gradient-Guided SA-TS Optimization. This is the most expensive stage, using Simulated Annealing (SA) and Tabu Search (TS) across $I \approx 3000 \cdot B$ iterations for $B \leq 4$. Initial population generation is $O(B \cdot P)$. Each iteration includes neighborhood operations ($O(P^2)$ worst-case), gradient-based evaluation ($O(P \cdot d)$), and tabu list updates ($O(\min(I, \text{tabu\_size}))$). Temperature scheduling affects cost: early iterations favor faster random operations ($O(P)$), while later ones may involve more structured edits ($O(P^2)$). Overall, this phase scales as $O(I \cdot P^2 \cdot d)$.

Gradient Analysis and Token Adjustment. This final phase refines token placements based on gradient similarity. It involves token-wise evaluation, correctness checks, and updates, all of which scale as $O(B \cdot P \cdot d)$.

Combined Complexity. Summing the above, the total cost is:

$$O(V \cdot d + I \cdot P^2 \cdot d + B \cdot P \cdot d) \approx O(V + I \cdot P^2 \cdot d)$$

assuming d and V are constant per model and $I \propto B$. Given that $P^2 \ll V$ (e.g., $P^2 = 900$, $V = 30{,}000$), EDR remains efficient for small B and moderate P. However, cost grows quadratically with sequence length and linearly with batch size, making EDR especially suitable for typical federated learning scenarios where B is small.

B Hyperparameter Tuning Details

Our parameter selection process followed a systematic approach.

Batch Size = 1. We conducted grid search experiments across the following ranges:

- **Initial temperature:** 100–500
- **Cooling rate:** 0.90–0.99
- **Maximum iterations:** 1,000–5,000

Through these experiments, we found that an initial temperature of 300, a cooling rate of 0.95, and 3,000 iterations provided the optimal balance between exploration and exploitation. These values yielded the best reconstruction quality without incurring excessive computational overhead.

Batch Size = 2 and 4. For larger batch sizes ($B = 2$ and $B = 4$), we followed a similar tuning process. Due to the expanded combinatorial search space, these settings required higher temperatures and more iterations to effectively explore the optimization landscape.

Specifically:

- For $B = 2$: Initial temperature = 400, cooling rate = 0.99, maximum iterations = 4,000
- For $B = 4$: Initial temperature = 500, cooling rate = 0.99, maximum iterations = 5,000

The increasing temperature and iteration values directly correspond to the greater complexity that must be explored as batch size increases. These adjustments ensured that the optimization process remained effective in various reconstruction scenarios.