Recovering Private Text in Federated Learning of Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Federated learning allows distributed users to collaboratively train a model while keeping each user's data private. Recently, a growing body of work has demonstrated that an eavesdropping attacker can effectively recover image data from gradients transmitted during federated learning. However, little progress has been made in recovering text data. In this paper, we present a novel attack method FILM[1] for federated learning of *language models*—for the first time, we show the feasibility of recovering text from *large batch sizes* of up to 128 sentences. Different from image-recovery methods which are optimized to match gradients, we take a distinct approach that first identifies a set of words from gradients and then directly reconstructs sentences based on beam search and a prior-based reordering strategy. The key insight of our attack is to leverage either prior knowledge in pre-trained language models or memorization during training. Despite its simplicity, we demonstrate that FILM can work well with several large-scale datasets—it can extract single sentences with high fidelity even for large batch sizes and recover multiple sentences from the batch successfully if the attack is applied iteratively. We hope our results can motivate future work in developing stronger attacks as well as new defense methods for training language models in federated learning.

1 Introduction

Federated learning [36] is a method to allow multiple participants to train a global model without exchanging their private data collaboratively. At each step of training, a central server transmits model parameters to every participating client. Each client then computes a model update (i.e., gradients) using its local data and sends it to the server. Finally, the server aggregates all updates—typically by averaging them—and updates the model. As each client does not need to share its private data with the server during training, federated learning is actively being considered for privacy-sensitive applications such as virtual mobile keyboards and analysis of electronic health records in hospitals [30].

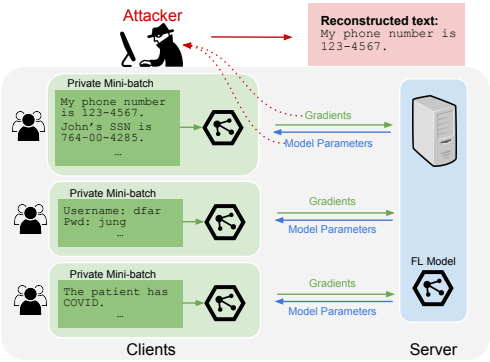

Figure 1: Illustration of the threat model and our attack FILM. FILM allows an *honest-but-curious* eavesdropper on the communication between the client and the server in federated learning to recover private text of the client.

However, recent studies [55, 16, 49, 54, 24] show that an *honest-but-curious* eavesdropping attacker in federated learning—who observes the communication of model parameters and gradients between a client and the server—can recover private data of the client (Fig. 1). By reformulating the attack as an optimization problem, an attacker can iteratively build a copy of a client's private images.

[1] FILM stands for **F**ederated **I**nversion Attack for **L**anguage **M**odels.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

However, the success in recovering text data remains limited—for instance, [55, 11] showed leakage of text data using similar optimization-based methods for only a batch size of 1[2].

In this work, we study federated learning of neural *language models* (LMs) and present a novel attack called FILM (**F**ederated **I**nversion Attack for **L**anguage **M**odels), which can recover private text data from information transmitted during training. For the first time, we demonstrate that an attacker can successfully recover sentences from *large* training batches of up to 128 sentences—making it practical and alarming in real-world scenarios.[3] We focus on language models for two reasons: (1) In the past few years, Transformer-based language models [48] have achieved unprecedented success [12, 41, 42, 7] and have been quickly adapted to a number of privacy-sensitive domains [2, 21, 32, 34, 29]. They are not only able to generate coherent and fluent text but serve as basic building blocks for language understanding tasks; (2) We identify that language-model training is at a higher risk of information leakage by nature because the attacker can possibly leverage the memorization capability of LMs during federated learning.

We find that previous gradient-matching approaches, which directly optimize in high-dimensional, continuous space, don't work well in recovering data for LMs as they are highly sensitive to initializations and language inputs are essentially discrete (Sec. 5). Especially with large batch sizes, the averaged gradient information (of ≤ 128 sentences) does not suffice to recover text information. Instead, our FILM attack takes a radically different approach: We first recover a set of discrete words[4] from the embeddings' gradients based on the insights from Melis et al. [37]. This step can give us a set of words that appear in any sentences of the training batch without the frequency information. We then develop a simple and effective strategy using beam search [43, 44], which tries to reconstruct one sentence from the set of identified words. This step is crucial as it takes advantage of prior knowledge encoded in pre-trained language models and memorization of training data during federated training. We also further design a token-reordering approach that leverages both language prior and gradients information as the signal to fine-tune the recovered sentence. Finally, we show that we may recover multiple sentences from the same training batch by applying the same procedure iteratively.

We evaluate FILM on the WikiText-103 dataset [38] and the Enron Email dataset [28], based on a GPT-2 model [42], with either pre-trained or randomly-initialized weights. We analyze the attack performance with different batch sizes, number of training data points, and number of training epochs. Our experiments demonstrate high-fidelity recovery of a single sentence feasible, and recovery of significant portions of sentences for training batches of up to 128 sentences. We also show the possibility to recover multiple sentences from the same batch ($1/3$ for a batch size of 16), by applying FILM iteratively. Together, we believe that our results raise serious privacy-leakage for federated learning in realistic scenarios and they should motivate a necessary redesign of defenses to provide meaningful privacy guarantees to clients for training language models in federated learning.

2 Related work

Data recovery from gradients. Gradients in federated learning have been shown to leak a severe amount of information of the underlying private data. Several attacks [55, 53, 16, 14, 49, 54, 26, 25] successfully reconstruct individual users' private images from gradients. These methods usually formulate such a reconstruction as an optimization problem, which encourages reconstructed images to yield similar gradients as observed gradients computed on the private images, using regularization terms based on image priors such as total variation [16] and batch normalization statistics [49].

Existing research has primarily focused on recovering images, and the earliest attempt to recover text from gradients is Zhu et al. [55], which briefly presented leakage results in masked language modeling [12]. Their attack aims to match target gradients with continuous representations and maps them back to words which are closest in the embedding matrix. Deng et al. [11] improve [55] by adding a regularization term which prioritizes the gradient matching in layers that are closer to the input data. However, both work only present results with batch size of 1 and do not work well with larger batch sizes, as we will show in Sec. 5. Another earlier work [37] shows the information leakage from the embedding layer by observing which entries of words are non-zero in the gradients—a key insight that we also build on in this work. However, different from their attack which only infers a person or location from a shallow GRU model [10], we are proposing a much stronger attack that

[2] We focus on *honest-but-curious* attackers in this work. If the attacker is allowed more capabilities (i.e., modifying model weights), it may achieve better performance (see Sec. 2).

[3] Another concurrent work [13] demonstrated the possibility of recovering a batch of 4 sentences on text classification tasks (see Sec. 2 for more discussion).

[4] Strictly speaking, we recover and reorder subword tokens [45] for modern neural language models. In this paper, we use 'words' in the description for simplicity.

aims at recovering full sentences from larger language models. Finally, very recent work [5, 15] tries to recover text data in federated learning under a stronger threat model: the server is malicious and can manipulate the training model's weights to enable easier text reconstruction.

The closest concurrent work is Dimitrov et al. [13], where the attacker exploits the prior probability of the private text by minimizing both the gradient matching loss and the text probability. They are able to recover a mini-batch of 4 sentences with binary classification tasks using the BERT [12] model. In this work, we focus on language models and we show that our attack is able to recover sentences from a larger training batch of up to 128 sentences.

Memorization in language models. Our method is also inspired by the work related to the memorization of training data in language models. Recent studies [8, 47, 46, 51, 9, 4] notice that large language models are able to memorize their training dataset and hence can be prompted to output specific sensitive information. For example, Carlini et al. [8] propose a way to quantitatively measure the risk that a training example is unintentionally memorized by a language model, and Thakkar et al. [47] extend this measurement to federated learning. Zanella-Béguelin et al. [51] demonstrate that an attacker who has access to the pre-trained and fine-tuned LSTM [18] is able to extract rare or unique sequences from the fine-tuning dataset. More recently, Carlini et al. [9] present an attack to extract personally identifiable information (e.g., name, email address, phone number, and physical address) from the publicly available GPT-2 model [42]. They also find that larger language models are more susceptible to this style of attack than smaller ones.

Note that while these previously demonstrated memorization attacks are trying to recover *a subset of* sensitive text from a large training corpora, our attack aims at recovering sentences from *every* mini-batch in federated learning. We stress that our attack is significantly more powerful than the previous works studying memorization.

We discuss further related work about defenses against data recovery from gradients in Appendix A.

3 Preliminaries

In this section, we describe relevant background of language models, federated learning, as well as the threat model of our proposed attack method.

3.1 Language Modeling

Language modeling is a core task in natural language processing and has been used as a building block of state-of-the-art pipelines [42, 12]. Given a sequence of n tokens $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, the language modeling task is to estimate the probability distribution of $\mathbb{P}(\mathbf{x})$, which is factorized as:

$$\log \mathbb{P}(\mathbf{x}) = \sum_{i=1}^{n} \log \mathbb{P}(x_i | x_1, ..., x_{i-1}). \tag{1}$$

where θ represent the neural network parameters. State-of-the-art neural language models are parameterized by RNNs [39] or Transformers [48], which consist of millions or billions of parameters. A language model first maps discrete words to a sequence of vectors via a word embedding matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $\mathcal{V}$ is the vocabulary and d is the hidden dimension. After computing the hidden representation $\mathbf{h}_i$ conditioned on $x_1, \ldots, x_{i-1}$, the model predicts the probability of the next token:

$$\mathbb{P}_\theta(x_i | x_1, ..., x_{i-1}) = \frac{\exp(\mathbf{h}_i^\top \mathbf{W}_{x_i})}{\sum_{j \in \mathcal{V}} \exp(\mathbf{h}_i^\top \mathbf{W}_j)}, \tag{2}$$

where $\mathbf{W}_{x_i}$ and $\mathbf{W}_j$ represent the word embeddings corresponding to x_i and j.

3.2 Federated Learning

Federated learning (FL) [36] is a communication protocol for training a shared machine learning model on decentralized data. An FL system usually involves N clients: $c_1, c_2, \cdots, c_N$ and a central server s, where the clients wish to collaboratively train a neural network f_θ (a language model in our case) with their private data $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_N$, under the coordination of the server. The training process optimizes θ (model parameters) using a loss function $\mathcal{L}$ and runs for T iterations.

Each iteration $t \in [T]$ of federated learning consists of the following steps. First, an individual client $c_i, i \in [N]$ computes $\nabla_{\theta^t} \mathcal{L}_{\theta^t}(\mathcal{B}_i)$, the gradients of the current model parameters θ^t on a randomly sampled private data batch $\mathcal{B}_i \subset \mathcal{D}_i$, $|\mathcal{B}_i| = b$, and shares the gradients with the server. The server then aggregates gradient updates from all clients and updates the model:

$$\theta^{t+1} = \theta^t - \eta \sum_{i=1}^{N} \nabla_{\theta^t} \mathcal{L}_{\theta^t}(\mathcal{B}_i) \tag{3}$$

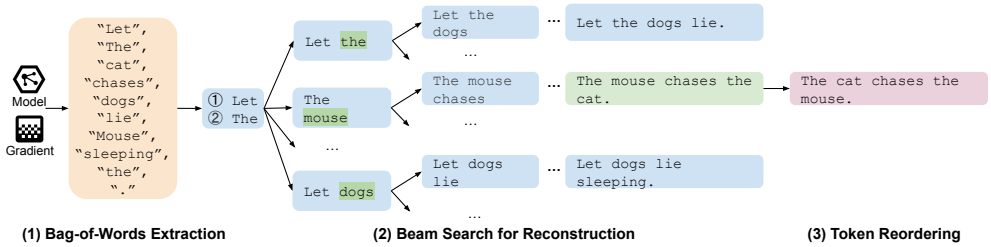

Figure 2: Illustration of our pipeline for batch size of 2. We firstly recover a bag of words from the observed gradients (orange square). We then select a small set of prompts and launch beam search. At every step of beam search, we append a new token from the bag of words and generate new states. We then order the states by their likelihood from the (trained) language model. We repeat the process until the generated sentences reach the length of the largest sentence in the batch. The state with the highest likelihood is selected as the beam search result (green square), and we reorder it to get the final result (purple square).

where η is the learning rate. Finally, the updated parameters θ^{t+1} are broadcast to individual clients. Note that Eq. 3 uses SGD for an easier demonstration for an update step in federated learning. In real-world training of neural language models, people usually use Adam [27].

In this work, we focus on federated learning of language models, where clients can start from a randomly-initialized neural network, or a pre-trained public model [17]—we consider a GPT-2 model [42] in our experiments.

3.3 Threat Model

Adversary's capabilities. We consider an honest-but-curious adversary who is eavesdropping on the communication between the central server s and an arbitrary client $c_i, i \in [N]$ in the federated training of a language model parameterized by θ, as described above. This allows the adversary a white-box access to 1) the gradients $\nabla_{\theta^t}\mathcal{L}_{\theta^t}(\mathcal{B}_i)$ sent by the client c_i, and 2) the model parameters θ^t (Fig. 1). Note that the adversary can inspect these information at *any* training iterations $t \in [T]$.

Adversary's objective. The adversary aims at recovering *at least one* sentence from the private training data batch $\mathcal{B}_i$ based on the information they observe, as recovering a single sentence is already sufficient to break the privacy guarantee of federated learning—since each training batch consists of randomly selected sentences from the private training dataset, if the adversary could recover at least one sentence from each training batch during the whole training procedure, he/she would be able to recover a considerable fraction of the original training dataset. Additionally, the attacker may also apply the attack on a single batch multiple times to recover more sentences from the batch (Sec. 4.5). The strength of an attack will be measured by the similarity between the recovered sentence and its corresponding private sentence in the original batch.

4 Our Attack Method

4.1 Overview

In this section, we present a novel text-recovery attack method, which is able to construct sentences from *any* training batches ($|\mathcal{B}_i| \leq 128$). Our method consists of three steps:

(1) Inspired by Melis et al. [37], our first step is to recover a set of words that may appear in a private batch ($\mathcal{B}_i$) from gradients (Sec. 4.2). While their goal is to identify sensitive single words from gradients, our approach is able to recover the order of words in a sentence.

(2) With the set of words identified, we then attempt searching for full sentences using beam search, based on the probability distribution provided by $f_\theta(x)$. Our method is able to leverage either the language prior from a pre-trained language model ($t = 0$), or the memorization capability of the language model during training ($t > 0$) (Sec. 4.3).[5]

(3) Finally, we design a prior-based scoring function and demonstrate that we can further improve the quality of recovered sentences by performing a simple and lightweight reordering step (Sec. 4.4).

We show that the above pipeline is able to recover single sentences from large training batches successfully. In Sec. 4.5, we discuss that we can iterate the above procedure to recover multiple sentences from the same training batch, which makes our attack even stronger.

In the following, we describe our attack method in detail. To better illustrate our attack's functionality, we provide a simplified step-by-step example that recovers a batch of 2 simple sentences in Fig.2.

[5]Recall that the attacker can inspect the gradients at any time step $t \in [T]$ during the T-epoch training. An attacker performing our attack at a larger t may have much stronger results than for a smaller t, as we later demonstrate in Sec. C.5.

4.2 Bag-of-Words Extraction

Input: $\nabla_{\theta^t}\mathcal{L}_{\theta^t}(\mathcal{B}_i)$, gradients of the language model on the private batch of sentences.
Output: bag of words B: {"Let", "The", "cat", "chases", "dogs", "lie", "mouse", "sleeping", "the", "."}.

Our first step is to extract a bag of words from $\nabla_{\theta^t}\mathcal{L}_{\theta^t}(\mathcal{B}_i)$, the gradients observed by an eavesdropper at the t-th iteration of training. We follow the method outlined by Melis et al. [37]—more specifically, we recover a set of words $B = \{w_1, w_2, \ldots, w_m\}$ by considering the non-zero rows of the embedding gradients $\nabla_{\mathbf{W}^t}\mathcal{L}_{\theta^t}(\mathcal{B}_i)$ ($\mathbf{W}^t$ is the word embedding matrix at training step t). Additionally, we are also able to recover the maximum length of sentences in the batch by analyzing the non-zero rows of the gradients corresponding to the positional embedding matrix.[6]

Note that this step only determines the set of words but not their *frequency*. Later, we will demonstrate that our beam search method can recover sentences well without knowing the frequencies (Sec. 6).

4.3 Beam Search for Sentence Reconstruction

Input: B, the bag of private words recovered by Sec. 4.2.
Output: a recovered sentence from the batch, "The mouse chases the cat.".

The second stage of our attack is to recover *one* sentence based on the extracted bag of words B and $\mathbb{P}_{\theta^t}(\mathbf{x})$ (note that this probability distribution is defined based on the parameters at the time of attack t). Since it is intractable to generate text directly using $\mathbb{P}_{\theta^t}(\mathbf{x})$, we use the beam search [43, 44], which is a greedy auto-regressive algorithm to generate sequences in a step-by-step manner with a trained language model. We provide a detailed algorithm in Appendix B.

To initiate beam search, we first find a set of words that can be chosen as starting tokens of sentences. A straightforward method is to select words which begin with a capital letter from B. For example, we would select "Let" and "My" as possible prompts for the illustrated batch. Then at each beam search step, we keep the top k possible beam states, finds the k most likely next words to each beam by using the language model (see procedure TOP($\cdot$) in Appendix B, Algorithm 1), and keeps the top k new beam states. The beam tree continues until the generated sentence reaches a specified length (we use 40 words), and outputs the text that has the overall highest probability (see Fig.2).

As the bag of words B does not contain information on the frequency of each word in the sentence, and beam search is prone to generating repetitive content [19], we also include a penalty for repeated n-grams in a sentence (see line 3 in procedure TOP($\cdot$)). We compare the performance of the attack under different n-gram penalties relative to a hypothetical optimal scenario where the frequency of each word is known by the attacker (implemented by sampling without replacement from the bag-of-words with word frequency information) in Sec. 6.

4.4 Prior-Guided Token Reordering

Input: a recovered sentence from Sec. 4.3, "The mouse chases the cat."
Output: a reordered sentence, "The cat chases the mouse."

By construction, beam search is a greedy method which generates sentences from left to right. Despite that it generally outputs fluent sentences by maximizing $\mathbb{P}_{\theta^t}(\mathbf{x})$, it does not leverage the *gradients* information which is the main signal in previous image attack methods. Thus, the final stage of our attack is to reorder tokens in the extracted sentence from the previous step, with an objective to improve its quality measured by a prior-based scoring function. The scoring function leverages both the perplexity and the gradient norm. Formally, given a sentence $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ of n tokens, and the trained language model parameterized by θ, $\mathbf{x}$'s prior score $\mathcal{S}(\mathbf{x})$ is defined as:

$$\mathcal{S}_\theta(\mathbf{x}) = \underbrace{\exp\left\{-\frac{1}{n}\log \mathbb{P}_\theta(\mathbf{x})\right\}}_{\text{Perplexity}} + \beta \underbrace{\|\nabla_\theta \mathcal{L}_\theta(\mathbf{x})\|}_{\text{Gradient Norm}} \tag{4}$$

where $\log \mathbb{P}_\theta(\mathbf{x})$ is the log-likelihood of the sentence (Eq. 1) according to the model parameters θ, and β is a hyper-parameter that controls the importance of gradient norm. Appendix C.6 provides empirical analysis that supports the design of this scoring function.

We also notice that sometimes the beam search preserves the structure of the sentence while the order of original phrases or words is wrong. Thus, we propose to reorder the sentences with two consecutive steps, from coarse to fine granularity.

[6] We assume that sentences in one batch vary in length. A standard practice is to pad shorter sentences.

Phrase-wise reordering. We first pre-process the sentence $\mathbf{x}$ by removing its redundant segments[7]. We then adjust the preprocessed sentence by reordering phrases inside it. At each reordering iteration, we generate multiple candidates by firstly cutting the sentence at p positions using the strategy in Malkin et al. [35], and then permuting the phrases between the cuts to form a new sentence. We then select the best candidate measured by $\mathcal{S}(\cdot)$. This stage terminates after 200 steps.

Token-wise reordering. We then reorder individual tokens. At each reordering step, we generate candidates using the following token-wise operations: 1) randomly swaps two tokens in the sentence, or 2) randomly deletes a token from the sentence, or 3) randomly inserts a token from the bag of words (recovered by Sec. 4.2) into the sentence. We then select the best candidate measured by $\mathcal{S}(\cdot)$ and move to the next iteration. This stage terminates after 200 steps.

4.5 Recovering Multiple Sentences

The previous steps only consider the recovery of a single sentence from a training batch. In this step, we describe an extension of our attack which is able to support recovery of multiple sentences from the same batch. We first perform beam search as described in Sec. 4.3. We store the results, and repeat beam search again—except this time applying an additional n-gram penalty based on the results of the previous search. Over many repetitions of this procedure, we are able to build a set of candidate recoveries for the original batch.

5 Experiments

5.1 Setup

Model and datasets. We evaluate the proposed attack with the GPT-2 base model [42] (around 117M parameters) on two language modeling datasets, including WikiText-103 [38] (license: CC BY-SA 3.0) and the Enron Email dataset [28] (license: unknown). Both datasets are publicly available for research uses. We choose WikiText-103 because it is commonly used in the language modeling task, and the Enron Email dataset because it consists of private email messages that contain abundant private information such as individuals' names, addresses, and even passwords. Note that the GPT-2 model is not trained on any Wikipedia data, and very unlikely on the Enron Email data[8].

We evaluate our attack on a subset of the WikiText-103 and Enron datasets. After pre-processing, we have 203,456 sentences from the WikiText-103 dataset and 31,797 sentences from the Enron dataset[9].

Training and attack settings. Unless otherwise noted, we train the model on these sentences for $90,000$ iterations using an initial learning rate of 1×10^{-5}, with a linearly decayed learning rate scheduler. All models were trained using early stopping—i.e., models were trained until the loss of the model on the evaluation set increased. We note that the running time of our algorithm is also quite fast. Our approach can recover a single sentence in under a minute using an Nvidia 2080TI GPU.

Evaluation metrics. We propose to use the following metrics to evaluate the attack performance: (a) Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [33] is a set of metrics for evaluating automatic summarization of texts as well as machine translation. We use ROUGE-1, ROUGE-2, and ROUGE-L F-Scores to evaluate the similarity between the recovered and the original sentences. More specifically, ROUGE-1 and ROUGE-2 refer to the overlap of unigrams and bigrams between the recovered and the original text, respectively. ROUGE-L measures the longest matching subsequence of tokens. For ablation studies we only show ROUGE-L as a high score is more representative of significant leakage than ROUGE-1 or ROUGE-2.

(b) Named entity recovery ratio (NERR) computes the percentage of original name entities that can be perfectly retrieved in the recovered text. Since named entities usually contain sensitive information (e.g., names, addresses, dates, or events), NERR measures how well the attacker recovers such information, ranging from a complete mismatch (NERR = 0) to a perfect recovery (NERR = 1). We identify named entities using the API provided by SpaCy [20].

[7] We notice that the sentences we recover from the previous step may not naturally finish after the beam search generates a punctuation. To address this problem, we pre-process each recovered sentence $\mathbf{x}$ by removing segments after its first punctuation, as long as the resulting sentence $\mathbf{x}'$ satisfies $\mathcal{S}(\mathbf{x}') < \mathcal{S}(\mathbf{x})$.

[8] The GPT-2 model is trained on a corpus that contains all web pages from outbound links on Reddit which received at least 3 karma [42]. Note that all Wikipedia pages were removed from this dataset.

[9] Both datasets have longer context and we preprocess them into sentences and use each sentence as a training instance. We leave recovering longer paragraphs as future work.

Attack & Batch Size b	Original Sentence	Best Recovered Sentence
	WikiText-103	
Zhu et al. [55]*, $b = 1$	As Elizabeth Hoiem explains, "The most English of all Englishmen, then, is both king and slave, in many ways indistinguishable from Stephen Black.	thelesshovahrued theAsWords reporting the Youngerselagebalance, mathemat mathemat mathemat reper arrangpmwikiIndia Bowen perspectoulos subur, maximal
FILM, $b = 1$	The short@-@tail stingray forages for food both during the day and at night.	The short@-@tail stingray forages for food both during the day and at night.
FILM, $b = 4$	More than 1@,@500 of Maruyama's troops were killed in the attacks while the Americans lost about 60 killed.	More than 1@,@500 of their troops were killed in the attacks, while the Americans lost about 60 killed on their own.
FILM, $b = 16$	A tropical wave organized into a distinct area of disturbed weather just south of the Mexican port of Manzanillo, Colima, on August 22 and gradually moved to the northwest.	Early on September 22, an area of disturbed weather organized into a tropical wave, which moved to the northwest of the area, and then moved into the north and south@-@to the northeast.
FILM, $b = 128$	A remastered version of the game will be released on PlayStation 4, Xbox One and PC alongside Call of Duty: Infinite Warfare on November 4, 2016.	At the time of writing, the game has been released on PlayStation 4, Xbox One, PlayStation 3, and PC, with the PC version being released in North America on November 18th, 2014.
	Enron Email	
Zhu et al. [55], $b = 1$	Don and rogers have decided for cost management purposes to leave it consolidated at this point.	dj "... Free, expShopcriptynt)beccagressive Highlands andinos Andrea Rebell impacts
FILM, $b = 1$	Volume mgmt is trying to clear up these issues.	Volume mgmt is trying to clear up these issues.
FILM, $b = 4$	Shares of enron traded as low as $15.	Shares of enron traded as low as $15 at jeffrey k.
FILM, $b = 16$	Yesterday, enron ousted chief financial officer andrew fastow amid a securities and exchange commission inquiry into partnerships he ran that cost the largest energy trader $35 million.	Yesterday, enron ousted its chief financial officer, andrew fastow, amid a securities and exchange commission inquiry into partnerships he ran that cost the company $35 million in stock and other financial assets.
FILM, $b = 128$	Yesterday, enron ousted chief financial officer andrew fastow amid a securities and exchange commission inquiry into partnerships he ran that cost the largest energy trader $35 million.	Yesterday, enron ousted chief financial officer andrew fastow amid a securities and exchange commission inquiry into partnerships he ran that he said cost the company more than $1 billion in stock and other assets.

* We also tried adding L1 regularizer in the objective (as suggested in Deng et al. [11]) and found similar attack results.

Table 1: Performance of Zhu et al. [55]'s attack with batch size = 1 (more in Appendix C.1) and FILM (ours) with different batch sizes. We show the best recovered sentence among 20 tested batches for each batch size (we report average results in Fig.3). Text in green represents the phrases and words that are recovered successfully.

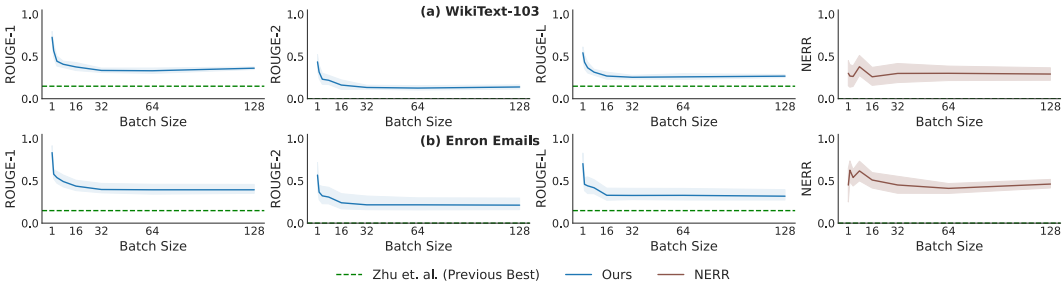

Figure 3: Recovery performance for various batch sizes on WikiText-103 (a) and the Enron dataset (b). Solid lines indicate the average F-Scores of recoveries out of 20 tested mini-batches. The attack is overall weaker for larger batch sizes (see Table 7 for quantitative results). Moreover, the attack is stronger for the Enron dataset.

5.2 Performance of Our Attack

Scaling to large training batches. We first compare the performance with respect to different batch sizes. In this setting, we start from a pre-trained GPT-2 model and fine-tune it on both datasets respectively (WikiText-103 and Enron emails). All models were finetuned for 90,000 iterations, at which the model loss over the evaluation set starts to increase. Fig. 3 shows the performance of our attack using different batch sizes with the WikiText-103 dataset and the Enron Email dataset. We observe that for a batch size of 1, all four scores (ROUGE-1, ROUGE-2, ROUGE-L, NERR) are close to 1, indicating a near-perfect recovery. We observe that as the batch size increases, the quality of the recovery by ROUGE-1, ROUGE-2, and NERR start to decrease. We attribute this drop in performance to the bigger set of words in the same training batch, as well as the growing search space of possible word orderings. We also observe that the Enron dataset is more susceptible to attack.

Table 1 shows the approach by Zhu et al. [55] fails to recover the original sentence for batch size $b = 1$. We find that the gradient matching method in Zhu et al. [55] is sensitive to initialization and it can perform better recovery when using a good initialization (see Appendix C.1). Fig. 3 shows their approach has poor NERR and ROUGE scores. On the contrary, our approach is able to recover a large portion of the original sentences on both datasets even when the batch size is 128. This demonstrates the superior effectiveness of leveraging the knowledge encoded in pre-trained language models and its memorization ability to recover sentences, compared to directly performing optimization in a high-dimensional continuous space.

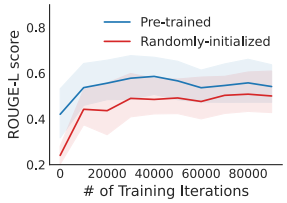

Figure 4: Attack performance over training iterations t for pre-trained or randomly-initialized model (batch size = 1).

Original: The short@-@tail stingray forages for food both at night and during the day.	
t	**The recovered sentence**
0	The, short for short-tail stingray at night during night day during day at day night at
10,000	The stingray forages for food at night both at both during food food shortages at short food
20,000	The short@-@tail stingray forages for food during the day and for the night.
40,000	The short@-@tail stingray forages for food both at night and during the day.

Table 2: An illustration of the recovered sentence with different training iterations t (batch size = 1) with a pre-trained model. Text in green represents the phrases and words that are recovered successfully. FILM recovers better sentences when t increases. See Appendix C.4 for results with a randomly-initialized model.

Attack performance at different training iterations. Previous attack on the image data suggest that a well-trained model is harder to attack [16], because its loss may converge to around zero, resulting in very small-magnitude gradients. However, we find well-trained models are more vulnerable to our approach because the model is able to memorize data during training.

We show the attack performance of our approach to the models at different numbers of training iterations t. Specifically, we vary the number of training iterations $t \in \{0, 10000, \ldots, 90000\}$ and fix the batch size $b = 16$ and dataset size $N = 200,000$. As shown in Fig. 4 and Table 2, a pre-trained model which has not been fine-tuned on client data can recover sentences with an average ROUGE-L score of 0.41, indicating that prior knowledge encoded in pre-trained models may help recover sentences. When the model is fine-tuned on Wikitex-103, the average score increases to 0.53 - an almost 30% increase. Our results suggest that leveraging the memorization in the model yields a more powerful attack than only using the language prior encoded in the pre-trained model. We note that we would expect stronger attack performance (especially for smaller training epochs) when attacking larger-capacity models, as stronger models can memorize training instances easily.We report more results for $t = 0$ with different batch sizes in Table 8 in Appendix C.3.

Pre-trained vs randomly-initialized models. We additionally compare the attack performance for training that starts from a pre-trained model (GPT-2 in our case) and a randomly-initialized model (we use the same GPT-2 architecture but re-initialize all the weights randomly). As shown in Fig. 4, we find that starting from a pre-trained model is more susceptible to attack than training from scratch. However, the gap in performance gradually decreases as the number of iterations increases.

Recovering multiple sentences by iterative attack. We further investigate a variant of our attack where we apply the beam search for multiple repetitions; For each repetition applying an n-gram penalty based on the results of previous generations. We consider a sentence to have strong leakage if the longest recovered subsequence is > 25% of the original sentence (i.e., ROUGE-L score > 0.25).

Table 3 shows that for a batch size of 16, an attacker who performs 100 repetitions of beam search is able to recover roughly 34% of the original batch (roughly 5 sentences) with ROUGE-L score over 0.25 (recall). However, since 100 repetitions results in 100 different generated sentences, an attacker would still need a method that picks good recoveries out of them[10]. We estimate the difficulty of choosing good sentences by using the ratio of good sentences to total generated sentences (precision). Table 4 shows a qualitative example of this iterative process.

Iterations	Recall	Precision
1	0.03 ± 0.03	0.50 ± 0.51
5	0.13 ± 0.08	0.36 ± 0.22
10	0.19 ± 0.11	0.28 ± 0.16
20	0.25 ± 0.12	0.21 ± 0.11
50	0.30 ± 0.11	0.12 ± 0.04
100	0.34 ± 0.09	0.07 ± 0.02

Table 3: Performance of recovering multiple sentences in a batch of size 16 by applying our attack iteratively. We measure the recall (% of original batch with ROUGE-L>0.25) and precision (% of recovered sentences out of all iterations with ROUGE-L>0.25).

Beam search iter	Original Sentence	Recovered Sentence
1st	In 1988 it was made into a feature film Katinka, directed by Max von Sydow, starring Tammi Øst as Katinka.	After the end of the year, the film was made into a feature film, and it was directed by Max von Sydow and published by Warner Bros.
2nd	He was killed in a mysterious plane crash on 29 March 1959, while en route to Bangui.	Fox, in an attempt to put an end to the fire, was killed in a plane crash while en route to Guadalcanal, Puerto Rico, on April 6, 2015, while on a
10th	The next year, Edwards published a significant treatise entitled Bibliographic Catalogue of the Described Transformation of North American Lepidoptera.	He had also published a treatise on it, published in 1959, entitled Bibliographic Catalogue of I.

Table 4: Qualitative results for multi-sentence recovery for a batchsize of 16. The multi-sentence attack is able to recover significant portions of multiple sentences in a single batch.

[10]In our experiments we use ROUGE-L score, but an attacker in a realistic scenario would not have access to this information.

6 Analysis

We finally discuss the impact of different parameters of our FILM attack in Fig. 5 and Fig. 6. We also study the effect of different training parameters and present the results in Appendix C.5.

Beam size. As discussed in Sec. 4.3, beam size (denoted by k) is a hyper-parameter of beam search which controls the number of beams active at any time. As shown in Fig. 5 (right), the attack performance grows as the beam size is increased. This result aligns with our intuition using a larger beam size corresponds with consideration of more sentences during search. Considering that the computational resources required for the attack also increases with a larger beam size, we find that a beam size of around 32 achieves the best trade-off in terms of attack performance and efficiency.

N-gram penalty. The n-gram penalty controls for what size of n-gram the penalty is applied. In Fig. 5 (left), we observe that having no penalty performs significantly worse than applying a penalty on repeated 2-grams. We note that performance for $n = 2$ (i.e., penalizing repeated 2-grams) is the closest to the "oracle" case, where the frequency of each word is known from the bag of words.

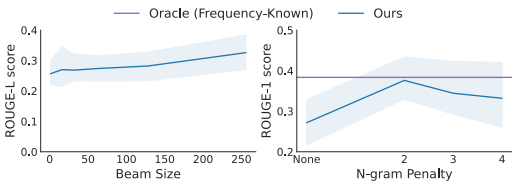

Figure 5: The effect of beam search configurations on our attack, including the n-gram penalty and the beam size. We note that an n-gram penalty with $n = 2$ performs the best. Recovery quality generally additionally improves with larger beam sizes.

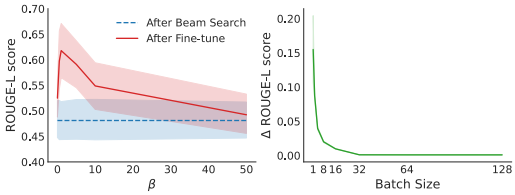

Figure 6: The effect of gradient norm coefficient β in the fine-tuning stage (see Section 4.4) with batch size of 1 (left), and the improvement of ROUGE-L score after fine-tuning for different batch sizes (right). We evaluate on the WikiText-103 dataset.

Gradient norm's coefficient β in prior-based scoring. As discussed in Sec. 4.4, the prior-based token recording for a sentence $\mathbf{x}$ uses a scoring function $\mathcal{S}(\mathbf{x}) = \text{Perplexity}(\mathbf{x}) + \beta \cdot \text{Gradient Norm}(\mathbf{x})$, where β is a hyper-parameter that controls the importance of the gradient norm term. Our results (see Fig. 6 (left)) show that the fine-tuning stage achieves a trade-off between fully perplexity-guided (i.e., $\beta = 0$) and almost fully gradient norm-guided (i.e., $\beta = 50$), with $\beta = 1$ being the optimal. The trend is consistent across different batch sizes.

We also notice that the improvement in the token reordering stage is most significant for small batch sizes (see Fig. 6 (right)). This is probably because that the beam search result for smaller batch sizes is usually closer to its corresponding ground truth sentence than for larger batch sizes, which increases the chance of being further improved by fine-tuning.

7 Discussion and Future Work

This paper presents the first attack that can recover sentences from gradients in federated learning of language models. Our method FILM works well with modern neural language models based on Transformers and large batch sizes (up to 128)—raising serious privacy-leakage concerns for federated learning in real-world scenarios. The key insight of our study is that an adversary can gain substantial power to reconstruct sentences by taking advantage of the knowledge memorized by trained language models.

Limitations. The results presented in this paper have several limitations. First, we haven't evaluated whether our attack can generalize to other natural language processing tasks, such as sentence classification. Second, we haven't tried our attack for batch sizes beyond 128 and our ablation study does not report results for batch sizes beyond 16. Third, we haven't evaluated how existing defenses work in terms of preventing our attack. We additionally leave open the question of selecting good sentences in the multi-sentence recovery scenario (Sec. 5.2).

Ethical considerations. While our intention is to raise the awareness of the privacy risks in the language modeling task in federated learning, from the attacker's prospective, FILM is also likely a step towards even stronger attacks. Therefore, we hope that this work can also motivate a necessary redesign of defenses to provide meaningful privacy guarantees to clients for training language models in federated learning.

References

[1] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016.

[2] Alsentzer, E., Murphy, J. R., Boag, W., Weng, W.-H., Jin, D., Naumann, T., and McDermott, M. Publicly available clinical BERT embeddings. *arXiv preprint arXiv:1904.03323*, 2019.

[3] Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

[4] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, Virtual Event Canada, 2021. ACM.

[5] Boenisch, F., Dziedzic, A., Schuster, R., Shamsabadi, A. S., Shumailov, I., and Papernot, N. When the curious abandon honesty: Federated learning is not private. *arXiv preprint arXiv:2112.02918*, 2021.

[6] Bonawitz, K. A., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. Practical secure aggregation for federated learning on user-held data. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[7] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[8] Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., and Song, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, 2019.

[9] Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, 2021.

[10] Cho, K., van Merriënboer, B., Bahdanau, D., and Bengio, Y. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, 2014.

[11] Deng, J., Wang, Y., Li, J., Wang, C., Shang, C., Liu, H., Rajasekaran, S., and Ding, C. Tag: Gradient attack on transformer-based language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3600–3610, 2021.

[12] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[13] Dimitrov, D. I., Balunović, M., Jovanović, N., and Vechev, M. Lamp: Extracting text from gradients with language model priors. *arXiv preprint arXiv:2202.08827*, 2022.

[14] Enthoven, D. and Al-Ars, Z. Fidel: Reconstructing private training samples from weight updates in federated learning. *arXiv preprint arXiv:2101.00159*, 2021.

[15] Fowl, L., Geiping, J., Reich, S., Wen, Y., Czaja, W., Goldblum, M., and Goldstein, T. Decepticons: Corrupted transformers breach privacy in federated learning for language models. *arXiv preprint arXiv:2201.12675*, 2022.

[16] Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. Inverting gradients–how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[17] Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[18] Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[19] Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

[20] Honnibal, M. and Montani, I. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. 2017.

[21] Huang, K., Altosaar, J., and Ranganath, R. ClinicalBERT: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*, 2019.

[22] Huang, Y., Song, Z., Chen, D., Li, K., and Arora, S. TextHide: Tackling data privacy in language understanding tasks. In *Findings of EMNLP*, 2020.

[23] Huang, Y., Song, Z., Li, K., and Arora, S. InstaHide: Instance-hiding schemes for private distributed learning. In *International Conference on Machine Learning (ICML)*, 2020.

[24] Huang, Y., Gupta, S., Song, Z., Li, K., and Arora, S. Evaluating gradient inversion attacks and defenses in federated learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[25] Jeon, J., Lee, K., Oh, S., Ok, J., et al. Gradient inversion with generative image prior. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 2021.

[26] Jin, X., Chen, P.-Y., Hsu, C.-Y., Yu, C.-M., and Chen, T. Catastrophic data leakage in vertical federated learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 2021.

[27] Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[28] Klimt, B. and Yang, Y. The enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*. Springer, 2004.

[29] Kraljevic, Z., Shek, A., Bean, D., Bendayan, R., Teo, J., and Dobson, R. MedGPT: Medical concept prediction from clinical narratives. *arXiv preprint arXiv:2107.03134*, 2021.

[30] Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[31] Li, X., Tramer, F., Liang, P., and Hashimoto, T. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[32] Li, Y., Rao, S., Solares, J. R. A., Hassaine, A., Ramakrishnan, R., Canoy, D., Zhu, Y., Rahimi, K., and Salimi-Khorshidi, G. BEHRT: transformer for electronic health records. *Scientific reports*, 10(1):1–12, 2020.

[33] Lin, C.-Y. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, July 2004.

[34] Liu, D. and Miller, T. Federated pretraining and fine tuning of BERT using clinical notes from multiple silos. *arXiv preprint arXiv:2002.08562*, 2020.

[35] Malkin, N., Lanka, S., Goel, P., and Jojic, N. Studying word order through iterative shuffling. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 10351–10366, 2021.

[36] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., et al. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282, 2017.

[37] Melis, L., Song, C., De Cristofaro, E., and Shmatikov, V. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 691–706. IEEE, 2019.

[38] Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

[39] Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, volume 2, pp. 1045–1048. Makuhari, 2010.

[40] Phong, L. T., Aono, Y., Hayashi, T., Wang, L., and Moriai, S. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 2018.

[41] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018.

[42] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[43] Reddy, D. R. et al. Speech understanding systems: A summary of results of the five-year research effort. *Department of Computer Science. Camegie-Mell University, Pittsburgh, PA*, 17: 138, 1977.

[44] Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, 2010.

[45] Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. In *Association for Computational Linguistics (ACL)*, pp. 1715–1725, 2016.

[46] Song, C. and Raghunathan, A. Information leakage in embedding models. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020.

[47] Thakkar, O., Ramaswamy, S., Mathews, R., and Beaufays, F. Understanding unintended memorization in federated learning. In *Proceedings of the Third Workshop on Privacy in Natural Language Processing*, pp. 1–10, 2021.

[48] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[49] Yin, H., Mallya, A., Vahdat, A., Alvarez, J. M., Kautz, J., and Molchanov, P. See through gradients: Image batch recovery via gradinversion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[50] Yu, D., Naik, S., Backurs, A., Gopi, S., Inan, H. A., Kamath, G., Kulkarni, J., Lee, Y. T., Manoel, A., Wutschitz, L., et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

[51] Zanella-Béguelin, S., Wutschitz, L., Tople, S., Rühle, V., Paverd, A., Ohrimenko, O., Köpf, B., and Brockschmidt, M. Analyzing information leakage of updates to natural language models. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020.

[52] Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[53] Zhao, B., Mopuri, K. R., and Bilen, H. iDLG: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[54] Zhu, J. and Blaschko, M. B. R-gap: Recursive gradient attack on privacy. In *International Conference on Learning Representations (ICLR)*, 2021.

[55] Zhu, L., Liu, Z., and Han, S. Deep leakage from gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] .
   (b) Did you describe the limitations of your work? [Yes] , see Sec. 7.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] , see Sec. 7.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] .
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] , please check the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] , see Sec. 5.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] , see Fig 3.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] , see Sec. 5.1.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] .
   (b) Did you mention the license of the assets? [Yes] , see Sec. 5.1 (unfortunately, there is no license information for the Enron Email dataset).
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] , we include our code in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] , see Sec. 5.1.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] , we noted in Sec. 5.1 that "the Enron Email dataset because it consists of private email messages that contain abundant private information such as individuals' names, addresses, and even passwords".
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] .
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] .
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] .