Contrast-Then-Approximate: Analyzing Keyword Leakage of Generative Language Models

Zhirui Zeng, Tao Xiang, *Senior Member, IEEE*, Shangwei Guo, *Member, IEEE*, Jialing He, *Member, IEEE*, Qiao Zhang, *Member, IEEE*, Guowen Xu, *Member, IEEE*, and Tianwei Zhang, *Member, IEEE*

***Abstract*— There is an increasing tendency to fine-tune large-scale pre-trained language models (LMs) using small private datasets to improve their capability for downstream applications. In this paper, we systematically analyze the pre-train and then fine-tune the process of generative LMs and show that the fine-tuned LMs would leak sensitive keywords of the private datasets even without any prior knowledge of the downstream tasks. Specifically, we propose a novel and efficient keyword inference attack framework to accurately and maximally recover sensitive keywords. Owing to the fine-tuning process, pre-trained and fine-tuned models might respond differently to identical input prefixes. To identify potential sensitive sentences for training the fine-tuend LM, we introduce a contrast difference score that assesses the response variations between a pre-trained LM and its corresponding fine-tuned LM. Following this, we iteratively fine-tune the pre-trained model using these sensitive sentences to minimize the disparity between the target model and the pre-trained model, thereby maximizing the number of inferred sensitive keywords. We implement two types of keyword inference attacks (i.e., domain and private) according to our framework and conduct comprehensive experiments on three downstream applications to evaluate the performance. The experimental results demonstrate that our domain keyword inference attack achieves a precision of 85%, while our private keyword inference attack can extract highly sensitive personal information for a significant number of individuals (approximately 0.3% of all customers in the private fine-tuning dataset, which contains 40,000 pieces of personal information).**

***Index Terms*— Language models, fine-tuning, sensitive keywords, personal information leakage.**

I. Introduction

THE rapid development of deep learning techniques in Natural Language Processing (NLP) has led to significant advancements in Language Models (LMs), making them fundamental to various NLP tasks, such as text classification [1], [2] and question answering [3]. Recent popular LMs, such as Google's BERT [4] and OpenAI's GPT family [5], are composed of multiple layers of Transformer blocks with millions of parameters [6]. Pre-training these LMs on massive text corpora collected from the Internet is a common practice [7]. Large-scale LMs can understand and generate fluent natural language [8], and minor parameter updates enable direct application to various downstream tasks. Pre-trained LMs can additionally be fine-tuned on small private datasets for domain-specific applications without incurring the high costs of training from scratch [4].

Pre-trained language models (LMs) have become a popular way to improve the utility of downstream tasks and applications. The pre-training and fine-tuning process simplifies the training of LMs, and their public availability has led to several proposed attacks [9], [10], [11], [12], [13]. These attacks have shown that large, pre-trained LMs may memorize and leak individual samples [12]. Two types of attacks are possible: *membership inference* [9], [10], [14], [15], which infers whether an individual is part of the training data, and *sample inference* [12], [16], [17], [18], [19], [20], which directly extracts information from the individuals included in the training data. The latter poses a greater threat to LMs than the former, as it enables the extraction of more sensitive information beyond just the inclusion of a given individual in the training data. However, previous research has concentrated on establishing a minimum level of memorization without distinguishing between leaked public and sensitive information. For instance, leaking common or highly duplicated phrases is not considered a privacy violation, whereas leaking sensitive keywords of individuals is [21]. We believe that a comprehensive study on the risk of sensitive keyword memorization in LMs is lacking.

In this paper, we focus on analyzing the leakage of sensitive keywords from the training data of LMs. Particularly, we concentrate on the fine-tuning process of LMs, in contrast to most of the previous attacks that target the pre-training process. This is because the extracted information from pre-training datasets is often crawled from the Internet. However, compared to the pre-training datasets, the small private dataset used for fine-tuning is usually confidential and contains massive sensitive information that should be kept private. For instance, an airline can fine-tune a text summary model from a pre-trained LM using customer feedback to enhance its customer service [22].

Multiple challenges arise when attempting to extract sensitive keywords during the fine-tuning process. First, one can easily download the pre-trained LM but usually only have

Manuscript received 11 October 2023; revised 1 March 2024; accepted 25 March 2024. Date of publication 22 April 2024; date of current version 8 May 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB3103500; and in part by the National Natural Science Foundation of China under Grant 62102052, Grant U21A20463, and Grant 62302071. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Muhammad Khurram Khan. *(Corresponding author: Shangwei Guo.)*

Zhirui Zeng, Tao Xiang, Shangwei Guo, Jialing He, and Qiao Zhang are with the College of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: swguo@cqu.edu.cn).

Guowen Xu and Tianwei Zhang are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798.

Digital Object Identifier 10.1109/TIFS.2024.3392535

TABLE I
TAXONOMY OF PRIVACY INFERENCE ATTACKS

Category		Index	Model		Settings		Prompt
			Pre-trained	Fine-tuned	Black-box	White-box	
Inference Attacks on Input Data		C. Song [14]	√		√	√	
		X. Pan [23]	√		√	√	
Inference Attacks on Training Data	Membership Inference Attacks	V. Shmatikov [9]	-	-	√		
		S. Yeom [10]	-	-	√		
		C. Song [14]	√		√	√	
		F. Mireshghallah [15]		√		√	
	Sample Inference Attacks	E. Lehman [19]	√			√	√
		S. Zanella [11]	√	√		√	
		N. Carlini [12]	√		√		√
		R. Panchendrarajan [16]		√	√		
		H. A. Inan [17]		√		√	√
		T. Vakili [18]	√			√	√
		J. Huang [20]	√			√	√
	Sensitive Keyword Attack	Ours		√	√		

black-box access to the fine-tuned LM. How to infer significantly sensitive information (i.e., keywords) from limited clues is a significant challenge. Second, instead of inferring a small part of keywords, we want to recover as many sensitive keywords as possible, thereby exacerbating the threat of the attack. However, this goal is difficult to accomplish without prior knowledge of the private dataset utilized in the fine-tuning process.

To address the challenges, we design a novel and efficient attack framework to infer sensitive keywords from the fine-tuning process. Our framework can perfectly match the black-box setting and does not require any prior information about the targeted fine-tuned LM. It entails three core steps: sensitive sentence identification, sensitive dataset construction, and keyword extraction. We have leveraged the fact that pre-trained and fine-tuned models may exhibit behavioral differences for the same input prefix, owing to the fine-tuning process. Therefore, we first capitalize on the memorability of the target LM, which was fine-tuned on the private dataset, and propose a novel contrast difference score to identify sensitive sentences by comparing the responses of the pre-trained and fine-tuned (target) LMs on the same inputs. To maximally recover sensitive keywords, we fine-tune the pre-trained LM with the identified sensitive sentences and iterate the above process to reduce the difference between the latest fine-tuned and target LMs, from which we construct the sensitive dataset. This process could bridge the gap between the target LM and the pre-trained LM, enabling us to identify more sensitive sentences. We finally infer the keywords of the fine-tuning dataset by identifying the keywords of the sensitive dataset. To ensure reproducibility, we employ standard keyword extraction techniques for various forms of keyword inference attack.

In our attack experiments, we utilize our framework to implement two types of sensitive keyword attacks: domain and private. For our pre-trained LM, we employ a representative LM GPT-2 [5] released by OpenAI. We conduct our attacks on three specific downstream applications that are fine-tuned using three different types of private datasets. Our comprehensive experiments and analysis demonstrate that our domain keyword inference attack can infer domain keywords with a precision of 85%. Furthermore, our private keyword inference attack can extract massive amounts of personal private information from over 120 users, accounting for approximately 0.3% of all customers in the private fine-tuning data. Our novel attacks that utilize the proposed contrast difference score for sensitive keyword selection can extract up to 4 times more personal private information than random selection and 100 times more than state-of-the-art attacks.

Our main contributions are as below:

- We initially investigate the risk of sensitive keyword memorization in LMs and accordingly introduce the keyword inference attack targeting the fine-tuning process.
- We propose a novel and efficient attack framework to accurately and maximally recover sensitive keywords.
- We implement domain and private keyword inference attacks on multiple downstream applications to evaluate the effectiveness of our attacks.

Paper Organization. The rest of the paper is organized as follows: Section II provides an overview of related works, while Section III introduces some basic concepts relevant to generative language models and their implementation in the attacks presented in Section V. Section IV outlines the threat model and the taxonomy of the attacks, and presents their definitions. In Section V, we present the details of our domain keyword inference attack and private keyword inference attack. Our experiment results are presented in Section VI, followed by conclusions in Section VIII.

II. RELATED WORK

Recent studies [12], [13], [14], [23] have exposed the vulnerability of LMs to privacy attacks. These attacks can be classified into two categories based on the target privacy information they intend to reveal, namely inference attacks on input and training data. Table I presents these attacks.

A. Inference Attacks on Input Data

LMs serve as common embedding models for extracting the representation of input sentences. However, some inference attacks [14], [23] have revealed that embedding codes store information about the input data, including sensitive personal information. For instance, [14] conducted a white-box and black-box attack, training an inversion model that receives a text embedding code and produces the private words within

the input sentence. Sometimes, attackers may focus on specific sensitive words appearing in the input data within the embedding. Meanwhile, [23] developed an attack model against embedding models, training it to detect sensitive information from the word sequence's embedding vector representation.

B. Inference Attacks on Training Data

Numerous studies have demonstrated that large language models memorize the training data and it is possible to infer sensitive details of the training data by assuming a reasonable attack model.

1) Membership Inference Attacks: Membership inference attacks [9], [10], [14], [15] aim to determine if a sample belongs in the training dataset and these attacks have been applied to target LMs. Reference [9] proposed a user-level membership inference technique to enable LM auditing and determine if one's data was part of the training data. Discovering that the training data's words and sentences are more similar to each other than those not used when training, [14] designed a similarity score for identifying membership. Moreover, [15] evaluated the model's memorization across various fine-tuning approaches using a threshold-based membership inference attack.

2) Sample Inference Attacks: Different from member inference attacks that have access to some private data and aim to judge whether the private data is used to train the model, sample inference attacks intend to extract sensitive training samples from a given pre-trained or downstream LM, which are more harmful and more difficult to conduct. Reference [12] primarily focused on generative LMs available to the public, using six metrics, such as perplexity, to establish and select sensitive training data from the target model. To generating more diverse samples from the LM, one of their text generation strategies are feeding a prompt derived from internet scraping data to the model. This sampling strategy ensures that they will generate samples that are similar in nature to the type of data GPT-2 was trained on. Reference [11] examined the information leaks occurring in LM updates and proposed the differential score metric to distinguish the variation between the LM snapshots before and after the update. Reference [16] tried to extract the private information of the training data for fine-tuning pre-trained LMs by modeling the behavioral changes between the pre-trained and fine-tuned LMs. Reference [17] developed two metrics to detect the user-level data leak by measuring the model's ability to produce unique sentence fragments from the training data. They use each sequence in the training data as prompt to run the model. Their study revealed that data leakage hinges on various factors and proposed some mitigation suggestions. Recently, studies have elaborated on the leakage of personal data in masked language models, particularly those trained using clinical BERT models [18], [19] or large language models [20]. All these studies test the model using generic prompts to fill in masked tokens that represent the private information they wish to recover.

Few studies have explored the memorization of private personal information through sensitive keywords. Such keywords can be highly sensitive and their leakage can be damaging. Therefore, in this paper, we concentrate on analyzing the leakage of sensitive keywords from training data. On the other hand, these attacks mainly focus on the process of model pre-training, thus mainly extracting privacy information from the training datasets of pre-trained models. These training datasets, however, are usually large corpora crawled from the public Internet, which has no access to particular and specific users' confidential information. Instead, we aim to analyze the fine-tuning process and extracting the sensitive information from these small fine-tuning datasets.

III. Preliminaries

We will first present a primer on generative language models, followed by an introduction to utilizing zlib compression and generative techniques to extract keywords from sentences.

A. Generative Language Models

Machine learning models that generate natural language are used in various applications, including automatic caption generation, language translation [24], and next-word prediction [25]. Generative language models are autoregressive and operate on a fixed set of known tokens, referred to as the model's vocabulary, denoted by $\mathcal{V}$. They model the probability of a sequence of tokens $x_1, \ldots, x_i$ as the product of the per-token probabilities conditional on their prefix $Pr(x_i|x_1, \ldots, x_{i-1})$:

$$Pr(x_1, \ldots, x_{i-1}, x_i) = \prod_{i=1}^{n} Pr(x_i|x_1, \ldots, x_{i-1}) \quad (1)$$

The Transformer neural network architecture [26] is used in state-of-the-art language models. During training, one objective is to maximize the negative log-likelihood of the language model predicting the next token in training sentences given a prefix [27].

Equation 1 provides a probability distribution over all tokens in the vocabulary $\mathcal{V}$, which can be represented as a tree with $|\mathcal{V}|$ branches at each level. To generate text, the tree is traversed iteratively using greedy decoding [28], top-k sampling [29], or a beam search algorithm [30], while conditioning the language model on all preceding tokens.

Pre-training and fine-tuning are essential steps in developing state-of-the-art generative language models. During pre-training, the model is trained on a significant amount of unlabeled text data using a self-supervised learning approach [4]. The objective of pre-training is to acquire general language representations that capture the underlying structure and patterns of natural language. Fine-tuning follows pre-training and involves using a smaller labeled dataset to train the model on a specific task, such as text generation or language understanding. During fine-tuning, the pre-trained model's weights are updated based on the labeled data of the specific task. This approach is known as transfer learning [31], as the model can leverage the pre-trained knowledge to improve performance on the new task.

The process of pre-training and fine-tuning has gained popularity because it enables developers to train language models using limited labeled data, which is often a significant bottleneck in various NLP applications. The pre-trained models

can capture diverse linguistic patterns and structures, allowing them to generalize better to new tasks with limited training data. Fine-tuning can then be performed on the specific task, resulting in a model that is accurate and efficient.

B. Zlib Compression

Zlib [32] is a software library utilized to compress data through the DEFLATE algorithm, found in gzip file compression program [33]. The Zlib entropy is a measure of the randomness and predictability of the compressed data, with low entropy indicating higher compression rates. To determine the entropy of data compressed using Zlib, a frequency table is constructed, counting the occurrence of each symbol in the uncompressed data. The probability of each symbol's appearance is then calculated by dividing its frequency by the length of the data. This process can be formalized using the following equation:

$$-\sum_{i=1}^{n} p(i)\log_2(p(i)) \tag{2}$$

where $p(i)$ is the probability of symbol i appearing in the uncompressed data, and n is the total number of symbols.

The formula calculates the information entropy or self-information for each symbol i. The sum of the information entropy for all symbols provides the overall Zlib entropy value. Text compressors can identify repetitive, meaningless, or uninteresting patterns [12]. To filter out such patterns, we suggest a straightforward approach using Zlib compression. We estimate the Zlib entropy of the text by computing the number of bits of entropy when compressed with Zlib compression.

C. Keyword Extraction

Extracting keywords from large, diverse datasets with many documents can be challenging. Traditional methods, such as rule-based or frequency-based approaches, may not be adequate. Instead, advanced techniques like generative extraction, which employ machine learning algorithms to identify relevant keywords based on statistical patterns and data relationships, are becoming more popular in fields like natural language processing and information retrieval. However, these methods can be computationally intensive and require substantial amounts of training data. In this paper, we use *KeyBERT* and *NER-BERT* as our keyword extraction models:

KeyBERT [34] utilizes BERT embeddings [35] and unsupervised learning to extract keywords and keyphrases. Four main steps are included as it works: (1)BERT-based embeddings: It uses the pre-trained BERT model to generate contextualized word embeddings for each sentence in the input text; (2)Sentence clustering: KeyBERT then applies clustering algorithms, such as k-means, to group similar sentences together based on their embeddings; (3)Maximal Marginal Relevance (MMR) scoring: After clustering, KeyBERT uses the MMR algorithm to score each sentence based on its relevance and diversity within each cluster; (4)Keyword extraction: Finally, KeyBERT extracts the most important keywords or phrases from the top-ranked sentences. KeyBERT provides an efficient and effective way to extract the most informative keywords or phrases from text data, making it a popular choice for tasks such as document summarization and content analysis.

NER-BERT [36] is a pre-trained model designed named entity recognition (NER). The approach integrates two powerful deep learning models, BERT and a conditional random field (CRF) and comprises four primary steps: (1) BERT-based embeddings: It first uses the pre-trained BERT model to generate contextualized word embeddings for each token in the input text; (2) Fine-tuning BERT: Next, it fine-tunes the BERT model for named entity recognition by training it on labeled data; (3) CRF: A CRF layer considering the dependencies between neighboring tokens is then added to improve the overall accuracy of named entity recognition; (4)Inference: Once the model is trained, it can be used to predict named entities in new input text. NER-BERT has achieved state-of-the-art performance on various named entity recognition benchmarks and is widely used in natural language processing applications such as information extraction, question-answering, and sentiment analysis.

IV. Threat Model

We consider the generative NLP scenario with the pretrain-then-finetune learning strategy, where the pre-trained generative LM is trained on a large amount of unlabeled raw data and then fine-tuned for downstream tasks with limited private data. This includes domains such as natural language generation, chatbots, content recommendation systems, and more, where fine-tuning on private data is prevalent. Figure 1 illustrates the architecture of our keyword inference attack, which comprises the following entities. (1) *Pretrained Generative Model* $\mathcal{F}$. Existing popular generative LMs are mainly Transformer variants, consisting of millions of learnable parameters. They are pretrained on extremely large corpus such as the English Wikipedia in a unsupervised mode [37], which allows the models to learn more general features of the language. The well-trained LM $\mathcal{F}$ can generate meaningful sentences when given prefixes. In particular, given the i-th word/token, $\mathcal{F}$ outputs the probability of the next token $\mathcal{F}(x_1, \ldots, x_i)$. With the softmax operation, the probability of the next token is:

$$Pr(x_{i+1}|x_1, \ldots, x_i) = \text{softmax}(\mathcal{F}(x_1, \ldots, x_i)) \tag{3}$$

In our experiments, we adopt the widely used GPT-2 as the pre-trained LM. (2) *Downstream User.* With the small private dataset of a specific domain $\mathbb{D}_p$, the downstream user fine-tunes the pre-trained LMs and obtains the fine-tuned LM $\mathcal{F}_p$ to enhance model performance on the corresponding downstream applications or fields. Since training a large-scale pre-trained LM are highly expensive, state-of-the-art well-trained LMs are usually downloaded from publicly available sources. Compared with the training data for pre-training, the fine-tuning data is likely to contain more sensitive and private information for the downstream applications.

A. Adversary Capabilities

We consider the fine-tuned generative LM as an oracle in this study, with the adversary only able to access the

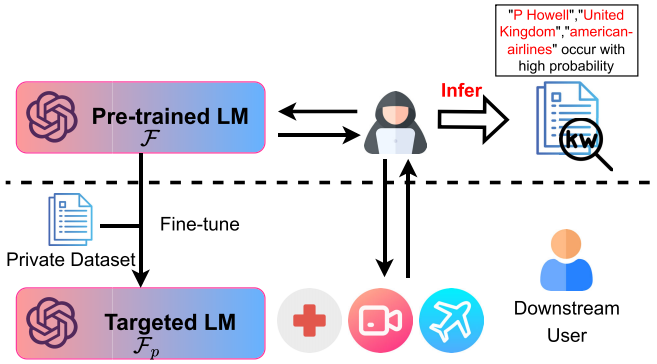


Fig. 1. Our keyword inference attack: we infer sensitive keywords with the black-box access of fine-tuned LMs.

target model through the black-box setting. In this study, we assume that the adversary has access to the pre-trained LM and can fine-tune it to some extent. The adversary can simultaneously query the pre-trained and target LMs to generate word sequences and their probability, but cannot examine the weights or hidden states of the target LM. Furthermore, we assume that the adversary has no prior knowledge about the fine-tuning dataset of the target model such as the distribution and the size of the dataset. This threat model is highly practical as various LMs can be queried by APIs in the black-box setting. For example, many models of the GPT family can be accessed in this manner. While certain models using users' datasets have implemented privacy protection measures, such as differentially-private training [38], [39], there is still a substantial risk of private information leakage when these models are released.

B. Attack Taxonomy and Goals

We consider two types of keyword inference attacks: domain and private. The *domain keyword inference attack* aims to infer the keywords of the finetuning dataset that contain related domain sensitive information. With the domain keywords, the adversary can profile the private fine-tuning dataset, which can further reconstruct a similar dataset for other malicious purposes. Alternatively, the *private keyword inference attack* enables the adversary to infer more specific user personal private information in the target dataset. Such an attack would seriously violate the privacy of data owners who voluntarily provide their private data for fine-tuning purposes.

1) Domain Keyword Inference Attacks: In our domain keywords inference attack, the adversary is primarily interested in inferring the domain-related keywords of the target dataset. Contrary to extracting precise sections of the training dataset, the goal of this attack is to indiscriminately extract the keywords of the target dataset that related domain sensitive information. For example, in the sentence "Women with early breast cancer (less than 3 cm) and no palpable axillary nodes were recruited", "breast cancer" is considered the domain keyword. By identifying the keyword, the adversary can infer that the private dataset is likely related to the medical domain, potentially involving surgeries or patient symptoms of breast cancer, which facilitates profiling the fine-tuning dataset. Thus, domain keywords can be highly sensitive, containing confidential and unpublished information.

2) Private Keyword Inference Attacks: In contrast to domain keyword inference attack, private keyword inference attack is more specific and targeted. The adversary is more curious about sensitive information of specific users, such as their name, location, and organization, contained in the fine-tuning dataset. This attack is much more challenging to conduct than the domain keyword attack. By launching a private keyword inference attack, the adversary can construct a library of personal sensitive information gleaned from the target model, which is extremely harmful. For example, an airline company wants to refine its customer service experience and employs customer reviews to fine-tune a pre-trained language model. If the adversary were to obtain customers' personal information contained in the reviews, such action would be deemed a severe violation of customers' privacy and even be dangerous to personal safety.

V. Keyword Inference Attacks

A. Overview

Although the pre-train and fine-tune paradigm provides a direct and effective way to fine-tune and apply state-of-the-art pre-trained generative LMs to the corresponding downstream tasks and applications, we find this process is accompanied by hidden privacy risks. We intend to analyze the fine-tuning leakage and infer sensitive keywords of the private fine-tuning dataset. Our key insights are two-fold. First, due to the memorability of the fine-tuned model, we filter out sensitive sentences by comparing the difference between the responses of the pre-trained and fine-tuned LMs. Then we continuously fine-tune the pre-trained LM to reduce the difference between the new fine-tuned and target LMs to maximize the extraction of sensitive keywords. We propose a novel keyword inference attack framework to extract sensitive keywords of the private fine-tuning dataset, which consists of the following three steps.

- *Sensitive Sentence Identification.* We generate a large number of sentences by unconditional sampling from the target model and then identify sensitive sentences from the candidates.
- *Sensitive Dataset Construction.* We next iteratively run the sensitive sentence identification process and use the filtered sensitive sentences to fine-tune the pre-trained model and merge all the sensitive sentences as the sensitive dataset.
- *Keyword Extraction.* We further identify and filter the specific keywords of the sensitive sentences in the sensitive dataset.

Figure 2 illustrates the pipeline of our keyword inference attack framework. Details of all three sentence identification, dataset construction, and keyword extraction steps follow shortly in this section.

B. Sensitive Sentence Identification

1) Candidate Sentence Generation: We first generate candidate sentences by continuously feeding $\mathcal{F}_p$ with an empty

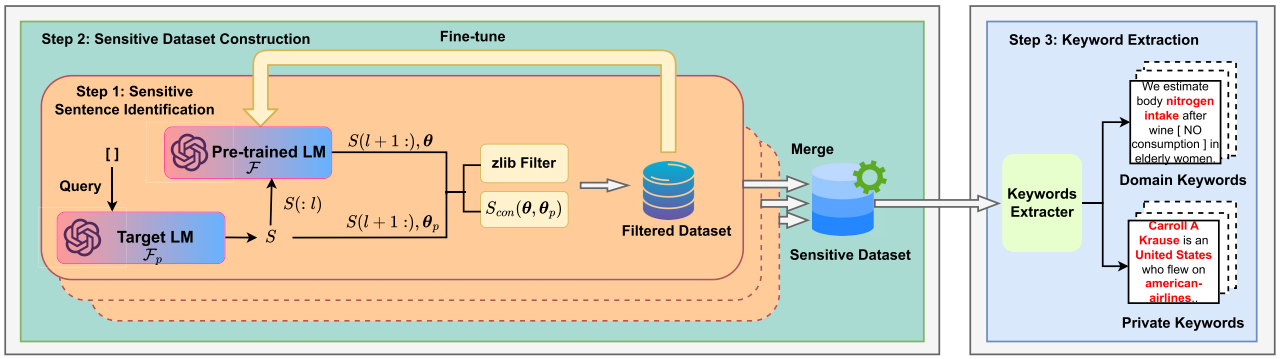


Fig. 2. Pipeline of our keyword inference framework.

input "[ ]". To increase the diversity of the generated sentences, we randomly sample a token from the output tokens with top k highest probability values (i.e., top-k sampling) [29] as the last token of the next query. Then, we can obtain a sentence as the candidate sentence by repeating the query process until $\mathcal{F}_p$ outputs the $[EOS]$ (end of the sentence) token or the sentence length reaches the pre-set bound.

Determining the optimal value of k poses a challenging task. A higher value of k results in $\mathcal{F}_p$ generating tokens with lower confidence, which could lower the likelihood of generating memorized samples. Nevertheless, a smaller k is known to narrow down the diversity of candidate sentences. Sentences with high diversity play a crucial role in maximizing our ability to extract the target model's memorized sensitive keywords.

For example, in our experiments, candidate sentences often contain numerous repetitive words when k is set to 20. To alleviate this issue, we implement a time-decay strategy to regulate the token selection process. When generating a sentence, we initially select a token at random from the top $k+t$ most probable tokens. Here, the variance parameter t acts as the offset. Increasing the offset value enhances the diversity of the model's output. However, maintaining a consistently high offset value throughout the sentence generation process may cause the sampling process to deviate from the memorized output. Therefore, we implement an offset that follows a time-decay function [12], starting at $t = 20$ and reducing to $t = 10$ within the first 10 tokens of the sequence. This strategy enables the language model to explore high-diversity tokens while still following high-confidence paths during sentence generation.

2) Sensitive Sentence Identification: We initiate the process by employing the target language model $\mathcal{F}_p$ to generate candidate sentences, followed by querying the pre-trained language model $\mathcal{F}$ with the prefixes of these candidate sentences to replicate the sentence generation process. Through this procedure, we acquire the candidate sentences along with their corresponding probability vectors. The identification of sensitive sentences is then facilitated by assessing the difference between the probability vectors of $\mathcal{F}$ and $\mathcal{F}_p$. Formally, for a given candidate sentence S, we query the pre-trained language models with the prefixes $S[:l]$ of S. Next, we utilize the strategy for the candidate sentence generation to select the next token and record the corresponding probability value of the selected token. This selected token is concatenated as a new prefix $S[:l+1]$, and the pre-trained language models are queried again with this new prefix of S. This process repeats until the candidate sentence S is fully generated. Then, we can obtain the probability vector of $\mathcal{F}$ to predict each token of $S[l:]$. Note that the probability vectors of $\mathcal{F}_p$ can be obtained during the generation of the candidate sentence. We denote the probability vectors as $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_p$.

To determine whether a sentence contains sensitive keywords (i.e., a sensitive sentence), we exploit the fact that $\mathcal{F}$ and $\mathcal{F}_p$ may have a behavioral difference for the same prefix due to the fine-tuning process. Thus, to quantify the such behavioral difference, we design a novel metric, Contrast Difference Score. Concretely, we use the Kullback-Leibler (KL) divergence [40] between two probability vectors $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_p$ as the contrast difference score.

Definition 1 (Contrast Difference Score): Let $\mathcal{F}$ and $\mathcal{F}_p$ be the pre-trained and fine-tuned LMs. S denotes the sentence that is generated by $\mathcal{F}$ and $\mathcal{F}_p$ with the same prefix of length l. $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_p$ are the probability vectors of $\mathcal{F}$ and $\mathcal{F}_p$ to generate $S[l:]$ and ω is the generated word. The contrast difference score of $\mathcal{F}$ and $\mathcal{F}_p$ on S is defined as

$$S_{con}(\boldsymbol{\theta}||\boldsymbol{\theta}_p) = \sum_{\omega} P(\omega|\boldsymbol{\theta}) \log \frac{P(\omega|\boldsymbol{\theta})}{P(\omega|\boldsymbol{\theta}_p)} \tag{4}$$

a) Selection rationale: The selsction of the contrast difference sore as a method for sensitive sentence identification in our study was driven by its efficacy and applicability. A higher KL divergence score indicates a lower similarity between two distributions and thus a higher contrast difference score indicates a lower similarity between two probability vectors, which illustrates how different $\mathcal{F}$ and $\mathcal{F}_p$ predict the tokens in the sentence. Table II displays an example of the contrast difference scores of two types of sentences, where $\mathcal{F}_p$ is fine-tuned from GPT-2 using the PubMed dataset [41]. The private sentences in the table are from the fine-tuning dataset $\mathbb{D}_p$, while the public sentences are randomly selected from the Internet. We can observe that the contrast difference score of the sentence from $\mathbb{D}_p$ is much higher than that from the Internet. To further investigate the validity and generality of the contrast difference score, in our paper, we randomly sampled about 4000 samples from the Pubmed dataset (i.e., the fine-tuning dataset) and the SST-2 dataset (a public dataset from the Internet recording movie reviews),

TABLE II
CONTRAST DIFFERENCE SCORES OF DIFFERENT TYPES OF SENTENCES

Type	Sentence	Score
Private Sentence	The aim of this study was to evaluate the efficacy, safety, and complications of orbital steroid injection versus oral steroid therapy in the management of thyroid-related ophthalmopathy.	2.8439
	Primary outcomes are domain specific self-efficacy , HIV related quality of life , and outcomes of health education.	2.2520
Public Sentence	This study aimed to assess the tolerance/hypoallergenic and efficacy of a thickened AAF (TAAF) in these infants.	0.1949
	Of the 86 infants randomized, CMPA with eHF intolerance was confirmed in 75 infants; all of them tolerated the allocated AAFs.	0.0874

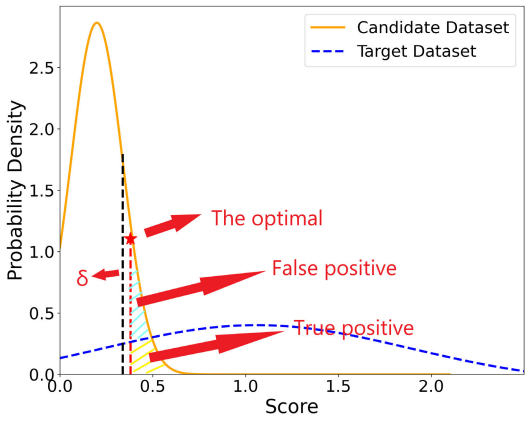


Fig. 3. An example that illustrates the reason for our threshold setting for sensitive sentence identification.

and then acquired the average of contrast difference scores as 1.05 and 0.28 respectively. This demonstrates the effectiveness of our contrast difference score to distinguish sentences originating from to sensitive target datasets. Experimental results on other different datasets (Pubmed, Rotten Tomatoes and Skytrax) demonstrate the generalization ability of our contrast difference score.

b) Addressing sensitivity: After computing the contrast difference score of a sentence, it is necessary to utilize a threshold value δ, to determine whether the sentence is sensitive enough, i.e., contains sensitive keywords. There is a potential risk associated with the sensitivity of the contrast difference score to variations. If the threshold value of contrast difference score is not appropriate, it might raise high false negative/positive rate. To mitigate this, we conduct a comprehensive analysis of the distribution of the contrast difference scores. The analysis reveals that the score distribution of both candidate sentences and target sentences follow a Gaussian distribution. Figure 3 illustrates the score distribution of generated sentences using the PubMed dataset. To minimize the impact of dataset size differences between the target dataset and our candidate dataset, we normalize the score distribution of candidate and target datasets. Observations show that only a relatively small number of sentences have contrast difference scores greater than 0.5. Therefore, the threshold value, δ, should be set in such a way that it minimizes both the false positive and false negative rates. We determine the mean μ and standard deviation σ of contrast difference scores of generated sentences and set $\delta = \mu + \sigma$. This selection ensures the elimination of massive low-sensitive sentences as well as the retention of an adequate number of high-sensitive sentences. As shown in Figure 3, the optimal value is at the point (marked with a star symbol) where the area of the false positive rate equals the area of the true positive rate (the red line intersects the scored distribution of sentences in PubMed dataset). The figure shows that our threshold closely approximates the optimal value.

To maximize the information content of the sentence candidates and minimize the generation of useless content such as "repeated" strings, we employ the zlib entropy [32]. Specifically, we calculate the zlib entropy of the sentence which represents the number of bits of entropy when the sequence undergoes compression using zlib compression. This method is useful for identifying sentences with repeated patterns and meaningless strings. By combining the contrast difference score and zlib entropy, we can filter out generated sentences that are less likely to contain sensitive keywords.

C. Sensitive Dataset Construction

Although we can identify sensitive sentences using the contrast difference scores, the corresponding sensitive keywords may be repeated and we cannot determine whether the number of sensitive keywords reaches the limits of our attack. To efficiently and maximally infer sensitive keywords, we propose to iteratively run the sensitive sentence identification process and fine-tune the pre-trained LM $\mathcal{F}$ using the filtered dataset at each iteration. The pre-trained LM can be continuously fine-tuned to reach the target LM $\mathcal{F}_p$ and follow its prediction probability. The newly identified sentences thus own lower and lower contrast difference scores in the iterative process and are helpful to reduce the repetition and maximize the extraction of sensitive keywords. This strategy isn't reliant on rigorous calibration to bypass overfitting. Actually, the objective of iteratively fine-tuning the pre-trained model is to accentuate the disparities between this model and the target model, from which we seek to derive keywords. Consequently, the performance of the pre-trained model doesn't hold primary significance in the context of our method and experiments. We finally merge all sentences in the filtered datasets as our sensitive dataset for further sensitive keyword identification.

D. Keyword Extraction

After constructing the sensitive dataset, we can use various detection methods to extract the sensitive keywords depending on what types of keywords we want. In the following, we implement the above two types of keyword inference attacks according to the proposed attack framework.

1) Domain Keyword Inference Attack: We detail our implementations for the domain keyword inference attack below. We first identify sensitive sentences and construct the sensitive dataset that contains domain keywords. Then, we use KeyBERT[1] to extract keywords from our sensitive dataset.

[1]https://maartengr.github.io/KeyBERT/index.html

We collect the extracted keywords to generate our domain keyword dataset for further inference.

2) Private Keyword Inference Attack: We detail our implementations for the private keyword inference attack below. `(1) Named entity extraction.` After identifying sensitive sentences and constructing the sensitive dataset, we use NER-BERT to extract the named entities sentence by sentence. Then, we collect the words of name, location, and organization in the sensitive dataset. `(2) Privacy information restoration.` After the named entity extraction, we use the name, location, or organization words as the inputs to query the target model and we can get the sentences that may contain more information such as country, date, and other private information. For example, 'P Howell' and 'K Nguyen' are first obtained in the name list. We then input these words as prefixes into the target model one by one and output the most likely next word each time, that is, to set top-k to 1. `(3) Privacy repository construction.` After restoring privacy information, we extract useful information from the private sentences to construct a private repository. To this end, one way is to extract relevant information based on the rules of the generated sentence format. More specifically, after the privacy information reconstruction process, most of the sentences we recover will have a similar structure. For example, we may obtain this kind of sentence: 'P Howell is a Germany who flew on adria-airways on 2015-04-10 and says Outbound flight ...'. Therefore, we can match and extract sensitive personal information based on this sentence pattern. Of course, not all sentences follow this fixed format. To maximize the extraction of sensitive keywords, we can thus exploit the other way, i.e., NERBERT to identify and extract other useful information based on the semantics of the sentence.

VI. Experiments

This section evaluates the proposed domain and private inference attacks from different aspects. Firstly, we introduce the common experimental setup, including the models and baselines used. Next, we present the specific configurations and experimental results of the domain and private inference attacks. The code and data used in this paper are available at https://github.com/zzrhh/keyword_inference_attacks.

A. Models

We evaluate our keyword attacks on LMs that are fine-tuned from pre-trained GPT-2 provided by OpenAI through the Huggingface Model Hub.[2] All experiments are conducted on LMs that are trained on the next-word prediction task and pre-trained on the WebText [5] dataset, which consists of 40GB of English text scraped from the Internet. GPT-2 uses a byte-pair encoder [42] for tokenization. More specifically, we conduct experiments with GPT-2 small ($124M$ parameters) and follow the common way of fine-tuning to fine-tune the entire model each time. We implement our keyword inference attacks in the black-box settings. We set an adaptive prefix length of sentences and the default value is set as the half length of each sentence, i.e., $l = \frac{|S|}{2}$ (1/2, for short), where $|S|$ denotes the length of the sentence S. We retain about 84% of the meaningful sentences using the zlib entropy.

B. Baselines

We implement the following keyword inference attacks as baselines.

- *Random*: the method that randomly selects the generated sentences from the target LMs for further keywords extraction;
- *Non-iteration*: the simplified version of our attack which does not use the filtered dataset to fine-tune the pre-trained LM in each iteration of the sensitive dataset construction.
- *Sample recovery* [12]: the attack exploiting the model's memorization ability to recover training samples of large pre-trained LMs. As a baseline in this paper, we transfer this attack to target the fine-tuning model.
- *Ours*: the attack proposed in Section V.

C. Domain Keyword Inference Attack

1) Configurations: We fine-tune GPT-2 with 100,000 sentences for each downstream application and generate 20,000 candidate sentences by querying each target LM in each iteration of the sensitive dataset construction. During the text generation process, the top-k sampling algorithm was utilized with a value of k set to 20 and an offset t (as described in Section V) set to 10. Furthermore, in each iteration, the model $\mathcal{F}$ will undergo 200 epochs of fine-tuning. For sensitive datasets, the KeyBERT model will be used to extract keywords, where each sentence corresponds to a keyword. In addition, we provide details of the other specific experimental setup, including the datasets and evaluation metrics used.

a) Datasets: The evaluation spans datasets from the medicine and movie domains: (1)*PubMed* [41] consists of approximately 200,000 abstracts of randomized controlled trials, totaling 2.3 million sentences. Each sentence of each abstract is labeled with their role in the abstract using one of the following classes: background, objective, method, result, or conclusion; (2)*Rotten Tomatoes*[3] consists of approximately 17,000 movies and their related critic reviews scraped from Rotten Tomatoes.

b) Metrics: We evaluate our domain keyword inference attack performance using Recall, Precision, and F1 score. Consistent with the attack definition, we specify members (i.e., the extracted keywords that appear in private datasets) as positive data points and non-members (i.e., the extracted keywords that disappear) as negative data points. Precision is the ratio of predicted correct keywords to all keywords in our constructed sensitive dataset, Recall is the ratio of predicted correct keywords to all keywords in the target dataset, and

[2]https://huggingface.co/gpt2

[3]https://www.kaggle.com/datasets/stefanoleone992/rotten-tomatoes-movies-and-critic-reviews-dataset

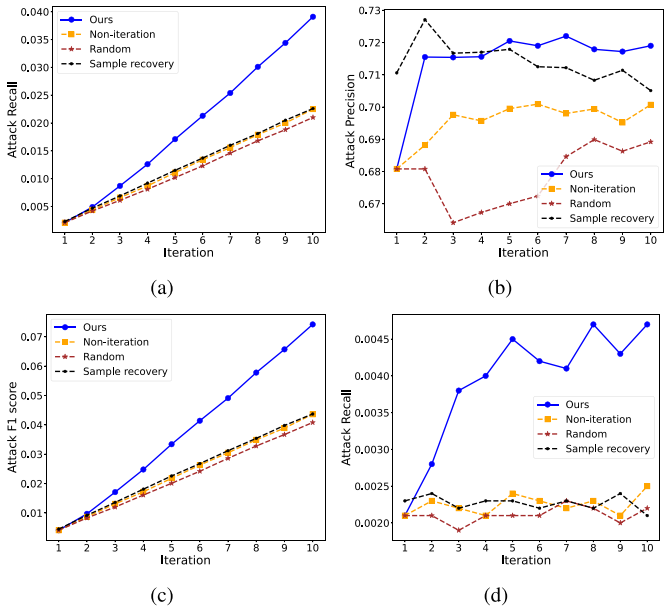


Fig. 4. Overall performance of our domain keyword inference attack on the medical dataset PubMed.

TABLE III
OVERALL EVALUATION OF OUR DOMAIN KEYWORD INFERENCE ATTACK

Method	PubMed			Rotten Tomatoes		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Random	68.92%	2.10%	4.08%	83.84%	3.15%	6.06%
Non-iteration	70.07%	2.25%	4.36%	82.16%	3.29%	6.32%
Sample recovery	70.51%	2.26%	4.37%	80.29%	3.21%	6.17%
Ours	**71.90%**	**3.91%**	**7.42%**	**84.98%**	**6.73%**	**12.48%**

F1 score [43] is the harmonic mean of precision and recall. Precision measures the attacker's confidence that a generated keyword also appears in the target dataset while Recall focuses on the number of sensitive keywords in the target dataset that are at risk of being extracted.

2) Evaluation Results: Using Precision, Recall, and F1 score, we first evaluate the overall performance of our domain keyword attacks against medicine and movie applications. The results are shown in Figure 4 and Table III. We also analyze the advantage of extracting phrases other than keywords from the datasets, as shown in Table V. Finally, we evaluate the effectiveness of different fine-tuning methods used, as depicted in Figure 5.

a) Overall evaluation: Figure 4 depicts the performance of our domain keyword inference attack on the medical dataset PubMed. In each iteration, we use KeyBERT to extract domain keywords from the sensitive dataset. We provide the corresponding Recall, Precision, and F1 score with respect to each iteration on our sensitive dataset, as shown in Figure 4 (a), 4 (b), and 4 (c), respectively. Meanwhile, the Recall of newly generated keywords in each iteration is also illustrated in Figure 4 (d) to demonstrate the advantage of our attack using fine-tuning operations to extract more sensitive keywords for each iteration. Among all baselines involved, our proposed attack attains the highest Recall, Precision, and F1 score in almost every iteration, especially after 10 iterations, our attack can gain a definite advantage. We owe the improved performances to using our contrast difference score to identify sensitive keywords. Moreover, the result in Figure 4 (d) demonstrates that our proposed method can outperform the baselines attaining the highest Recall to extract new keywords in iteration. This is achieved by using the obtained filtered dataset to fine-tune the pre-trained LM.

TABLE IV
OVERALL EVALUATION OF JAILBREAK ATTACKS

Method	Precision	Recall	F1 score
RWA	61.86%	3.37%	6.38%
AIM	63.15%	3.44%	6.52%
Base64	40.14%	2.18%	4.14%
PI	66.90%	3.64%	6.91%
Zulu	37.08%	2.02%	3.83%
Ours	**71.90%**	**3.91%**	**7.42%**

Table III further demonstrates the specific results of Precision, Recall, and F1 scores among three methods after the final iteration. It shows that our constructed sensitive dataset contains more domain keywords than the ones derived from the baselines. Our proposed method, which utilizes the contrast difference score to select sensitive sentences, achieves an 3.34% improvement in F1 score compared to the baseline approach of random selection (Random). Meanwhile, the proposed strategy that exploits the filtered dataset to fine-tune the target model can obtain a 3.06% gain compared to not doing so (Non-iteration). Furthermore, our method performs better than the existing attack against pre-trained LMs (Sample recovery), which attains an improvement of 3.05% in the F1 score. More specifically, Our pipeline runs for only 10 iterations, after which we obtain a sensitive dataset containing 4,651 sentences. We can extract an equal number of keywords from these sentences. In comparison, our training data includes 85,292 sentences (excluding some meaningless characters). Consequently, although our precision can reach 100%, our recall is constrained to 5.45% (4561/85292). Thus, when dealing with a large training dataset, achieving a high recall in a limited number of iterations is challenging. However, as depicted in Figure 4 (b), our method maintains a stable precision as the number of iterations increases. Therefore, if the training dataset comprises fewer sentences or if we run more iterations, we can achieve a higher recall without sacrificing precision, which we believe is more critical in a black-box setting for keyword extraction. This ensures confidence in the extracted keywords. Additionally, after 10 iterations, our recall is 3.91%, which is already very close to the maximum value (5.45%) of recall. We also observe similar results on Rotten Tomatoes, which further illustrates the versatility and effectiveness of our attack.

Moreover, we also conducted experiments related to jailbreak attacks [44], which aim to manipulate the large language model to generate unintended responses. Our comparative experiment with jailbreak attacks (refer to Table IV) substantiates the effectiveness of our attack. Specifically, we utilized five mainstream jailbreak methods (AIM, Base64, PI, Zulu and RWA) in large language models (LLMs) for comparison.

- *AIM*: An unfiltered and amoral prompt [45].
- *Base64*: This involves obfuscating the prompt through Base64 encoding to bypass the safety mechanisms of LLMs [46].

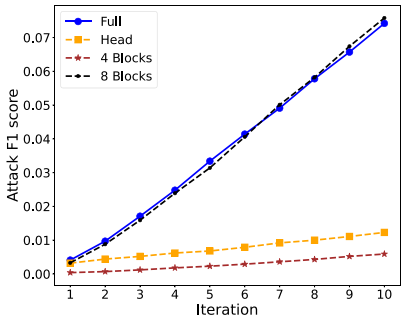


Fig. 5. Impact of the finetuned methods on the performance of our domain inference attack.

- *PI (Prefix Injection)*: This method requires LLMs to start the answer with a specific prefix [46].
- *Zulu*: This refers to using a low-resource language for jailbreak attacks [47].
- *RWA*: Repeat Words Attack.[4]

Table IV clearly demonstrates that the jailbreak attacks did not yield results as satisfying as ours. This comparison accentuates the superiority of our approach over traditional jailbreak methods.

b) The advantage of extracting phrases: On the other hand, we also use the KeyBERT to extract phrases other than individual words as keywords to show the effectiveness of our attack. In this case, our attack is able to obtain more sensitive information, which could pose a more serious threat to the target model. Table V displays the extracted phrases from the sensitive dataset using our method, along with the target sentences they appear in. We observed that the extracted keywords are highly informative as they contain detailed information about symptoms, drugs, and the hospital. Disclosing such information can lead to significant information leakage. For instance, if we only extract the keyword 'eyes', we will only know that the text pertains to eyes. However, if we extract the phrase 'dry eye', we can deduce that this text belongs to the medical domain, and could possibly be from a private dataset that contains information regarding patients with dry eye symptoms. It is essential to note that dry eye is a common symptom, however, if the target dataset comprises rare and sensitive diseases (such as AIDS) or drugs that are still undergoing clinical trials, the extraction of such information by an adversary can significantly violate a patient's privacy and lead to severe consequences.

c) Impact of fine-tuning methods: We also investigate the impacts of different fine-tuning methodologies in our domain keyword inference attack. GPT-2 model has 12 blocks, we thus apply four different settings to fine-tune using the first block(head), block 8-12(4 blocks), block 4-12(8 blocks), and all blocks(full) respectively, which has covered common fine-tuning approaches. Figure 5 shows the corresponding results. It can be seen from the figure that fine-tuning the entire model or just eight blocks of the model results in similar performance. However, when fine-tuning a smaller number of blocks(head or 4 blocks), our domain keyword inference attack becomes less effective.

[4]https://www.zdnet.com/article/chatgpt-can-leak-source-data-violate-privacy-says-googles-deepmind/

TABLE V
COMPARISON BETWEEN THE SENTENCES OF PUBMED AND THE SENSITIVE DATASET IDENTIFIED BY OUR DOMAIN KEYWORD INFERENCE ATTACK

Target Dataset	Sensitive Dataset
Although **ultrasound therapy** is used to treat calcific tendinitis of the shoulder, its efficacy has not been rigorously evaluated.	We assess personalized **ultrasound therapeutic** manucgate according the CT method when developing specific preferences.
The retention of calcium , phosphorus , sodium , potassium , copper , and zinc also did not differ between groups , and **nitrogen intake** did not affect mineral retention .	We estimate body **nitrogen intake** after wine [NO consumption] in elderly women.
For **children vaccinated** with a fourth dose of DTaP , which was the same DTaP as received in the primary series , fever and injection site redness , swelling , and pain increased in prevalence compared with the third dose in the primary series .	For **children vaccinated** by ELEN or ME I am confident we could demonstrate good sensitivity during immunocorruptive treatments by recording occurrence or adverse CNS or ASD reactions as defined (ESwSIDS - 4) along PTDI exposure thresholds.
The maximum **resting pressure** in the anal canal is greatly raised after hemorrhoidectomy.	While most men (N 1 940 women mean + LGBt + SD x hrs) were LIFTD individuals formatted towards anterior, this new trial group preferred TOFS with 30 kg excretion **resting pressure** : 76 + / 005 to 84 54 - kg.

TABLE VI
IMPACT OF THE KEYWORD EXTRACTION METHODS ON THE PERFORMANCE OF OUR DOMAIN KEYWORD INFERENCE ATTACK

Method	Precision	Recall	F1 score
TextRank	68.26%	3.69%	7.01%
Tfidf	74.66%	4.06%	7.71%
Yake	75.15%	4.08%	7.75%
KeyBert	71.90%	3.91%	7.42%

d) Impact of keyword extraction methods: We also investigate the impacts of different keyword extraction methods in our domain keyword inference attack. We compared our keyword extraction methods with some mainstream methods. TextRank [48] is an algorithm based on PageRank [49], which is often used in keyword extraction and text summarization. Term Frequency - Inverse Document Frequency (TF-IDF) [50] is a widely used statistical method in natural language processing and information retrieval. It measures how important a term is within a document relative to a collection of documents. Yake [51] is a light-weight unsupervised automatic keyword extraction method which rests on text statistical features extracted from single documents to select the most important keywords of a text. Table VI shows the corresponding results. We noted that the selection

TABLE VII

SAMPLES OF RECONSTRUCTED SENSITIVE LIBRARY BY OUR PRIVATE KEYWORD INFERENCE ATTACK

Name	Country	Airline	Date
L Arseniekowski	-	adria-airways	2006-08-28
Silkeno Guzenchmidt	United States	alaska-airlines	2012-07-15
P Howell	United Kingdom	american-airlines	2014-07-29
K Nguyen	United States	bangkok-airways	2011-07-29

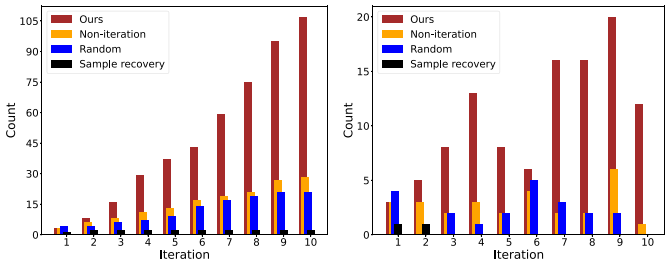


Fig. 6. Overall performance of our private keyword inference attack on the airline dataset.

TABLE VIII

THE NUMBER OF PASSENGERS' NAMES ASSOCIATE WITH AIRLINES, DATES OR COUNTRIES EXTRACTED BY OUR PRIVATE KEYWORDS INFERENCE ATTACK ON THE AIRLINE DATASET

Setting	Ours	No contextual information
Name	75	170
Name + $[airline, date, country]^1$	50	3
Name + $[airline, date, country]^2$	3	0

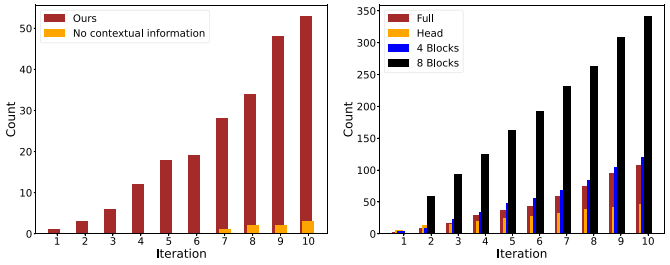


Fig. 7. Personal entity extraction and fine-tuning blocks comparison on airline dataset.

of the keyword extraction model did influence the results to a certain degree. Our keyword extraction methods (KeyBERT) exhibited compelling performance across this attack. Importantly, the selection of the keyword extraction method did not significantly influence our final results.

D. Private Keyword Inference Attack

1) Configurations: For our private keyword inference attack, our target dataset contains about 40,000 sentences. For each iteration in the sensitive dataset construction, we generate 10,000 candidate sentences by querying each target LM. We use the top-k sampling algorithm with k set to 20 and t set to 10, similar to the domain keyword inference attack. Additionally, we fine-tune the model $\mathcal{F}$ for 200 epochs. To identify name, location, and organization information in the sensitive dataset, we utilize NER-BERT during the keyword extraction process. We then use top-1 sampling to generate related sentences and extract relevant information using NER-BERT and grammatical rule-based methods. This information is used to build our privacy repository. We next provide details about the dataset and the evaluation metric used.

a) Dataset: The evaluation is conducted against an airline dataset *Skytrax*.[5] It is a scraped dataset created from all user reviews found on Skytrax which consists of 41396 Airline Reviews.

b) Metric: In particular, for the private keyword inference attack, as we are extracting the private information of a specific individual, we use the extracted user's name as a keyword to identify whether the individual appears in the target training dataset and record the number of correctly extracted keywords as a metric to evaluate the performance of this attack.

2) Evaluation Results: With the extracted private keywords, we can build a quadruple sensitive library ([name, country, date, airline]) (see Table VII). In the end, we have constructed a sensitive library including around 5,000 pieces of private information of individuals. We compare our built library and the target dataset and pick up the individuals that appear in our library and a target private sentence at the same time, the specific counts are delineated in Figure 6. We further analyze the advantage of using the extracted named entities to query the target model for further information recovery (as shown in the 'privacy information restoration' step of our private keyword inference attack). This essentially means that we can obtain additional contextual information by inputting the initially extracted named entities into the target model. Table VIII and Figure 7 (Left) list the corresponding number of correctly extracted private keywords using given contextual information and not. We finally evaluate the impacts of different fine-tuning methods used, as delineated in Figure 7 (right).

a) Overall evaluation: The target dataset and our constructed sensitive library typically contains multiple records for an individual passenger. Therefore, to fairly assess the effectiveness of our proposed approach, we only consider one relevant piece of information for each passenger when counting the number of correctly predicted keywords.

Figure 6 reports the performance of our private keywords inference attack on the airline dataset. As shown in Figure 6 (Left), the performance of our private keyword inference attack is much better than the method without fine-tuning (i.e., Non-iteration), random selection (Random) and existing attack method (Sample recovery). After 10 iterations, more than 100 passengers' personal information can be recovered using our attack, which is almost 100 times that of the Sample recover's result, and it is also much larger (4 times) than the number that Random and Non-iteration attain. Moreover, the number of newly generated keywords in each iteration is also illustrated in Figure 6 (Right) to demonstrate the advantage of our attack using the fine-tuning operation for each iteration. The existing attack method((Sample recovery)) that mainly

[5]https://github.com/quankiquanki/skytrax-reviews-dataset

focuses on pre-trained LM seems cannot directly apply to extract personal private information as its performance is even inferior to the Random and Non-iteration. Sample recovery only acquires few new keywords in each iteration or even fails to extract new keywords in some rounds. On contrary, our attack using the fine-tuning operation in each iteration can maximally and effectively extract sensitive information, which constantly extracts more new keywords in each iteration compared to the baselines.

b) The advantage of using contextual information: We have also discussed the difference between extracting contextual information from the target model to aid in subsequent information recovery and directly extracting keywords in the initial keyword extraction stage. In other words, we have discussed the difference between utilizing the 'privacy information restoration' step and not doing so. Intuitively, utilizing the target model to gather contextual information for future information recovery can increase the likelihood of extracting additional information about an individual, such as their country, airline, and date from the target airline dataset. In this experiment, we recorded the number of individuals correctly extracted with and without using contextual information, and the results are presented in Table VIII. Specifically, we exploit three settings with *Name*, *Name* + $[airline, date, country]^1$ and *Name* + $[airline, date, country]^2$. The setting *Name* means that we only extract names from the target dataset. The setting *Name* + $[airline, date, country]^1$ means that in addition to the passengers names, we also extract one of the pieces of information from the list [airline, date, country] associated with the names contained in the target dataset. Similarly, the setting *Name* + $[airline, date, country]^2$ means that we extract two pieces of information from [airline, date, country] that are associated with the name in the target dataset. Table VIII shows a significant difference between using and not using contextual information. The direct extraction method may be suitable for extracting a single keyword (name), as it has successfully extracted 175 passenger names from the target airline dataset. However, it is incapable of extracting more relevant information as it can only acquire 3 records with Name + $[airline, date, country]^2$. In contrast, our method using the target model to obtain contextual information achieves a significant improvement, as it extracts 50 individuals with Name + $[airline, date, country]^2$ records. In real life, obtaining more associated information for a passenger can pose a greater threat to their privacy. Thus, our attack, which utilizes the target model to acquire contextual information, can be even more harmful to personal information. Figure 7(Left) shows the total counts of correctly extracted individuals with the name plus two or more pieces of information from [country, date, airline]. It also illustrates that using the target model to obtain contextual information can facilitate the retrieval of more relevant private information for the target individuals.

c) Impact of fine-tuning methods: To have a full analysis of fine-tuning leakage, we also investigate the impacts of different fine-tuning methodologies in our private keyword inference attack. The same as the impact of domain keyword inference attack, we also apply four different settings to fine-tune using the first block(head), block 8-12(4 blocks), block 4-12(8 blocks), and all blocks(full) respectively. Figure 7(Right) shows the corresponding results. It is obvious that fine-tuning 8 blocks will leak more private dataset information to our private keyword inference attack while only fine-tuning the first block (head) would make our attack less effective.

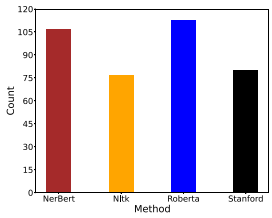


Fig. 8. Impact of the keyword extraction methods on the performance of our private keyword inference attack.

TABLE IX
OVERALL EVALUATION OF THE NUMBER OF QUERIES TO THE FINE-TUNED MODEL

Queries(k)	Step 1& Step2(GPU-days)	Step 3(ms)	F1 score(%)
10	0.04	33	4.41
15	0.06	37	6.18
20	0.07	41	7.42
25	0.11	42	6.68
30	0.13	39	7.35

d) Impact of keyword extraction methods: We also investigate the impacts of different keyword extraction methods in our private keyword inference attack. Similarly to the domain keyword inference attack, we selected another three different methods to conduct our experiment. NLTK [52] is widely used in academia and industry for name entity recognition. RoBERTa [53] builds on BERT and modifies key hyperparameters. Stanford NER [54] is a tool provided by the Stanford NLP Group that is used for identifying named entities in text. Figure 8 shows the corresponding results. It can be seen from the figure that our keyword extraction methods (NERBERT) exhibited compelling performance across this attack. Similarly to the domain keyword inference attack, the selection of the keyword extraction method did not significantly influence our final results.

VII. DISCUSSION

In the previous evaluations, our attack has demonstrated its effectiveness as well as its strong generalization ability. Furthermore, to obtain a comprehensive understanding of our keyword inference attacks, in this section, we focus on explaining the practicality of our threat model and the profound privacy implications brought by our attacks. The details are demonstrated as follows:

A. The Practicality of Our Threat Model

Our proposed attack model relies on the assumption of having access to the pre-trained model and the ability to fine-tune it multiple times. This threat model is highly pragmatic as various pre-trained language models such as Bert and GPT, are readily available. Fine-tuning these models

for downstream tasks, such as text generation, translation, and text classification, is straightforward and requires few computational resources. For example, a relatively simple task (e.g. text classification) with a moderate dataset might be achievable on a single high-end GPU or a cloud-based GPU instance [4].

To realize our attacks, we present the details on time complexity. In the following parts, we denote the length of a sentence s as $L(s)$, the total number of generate sentences as n, and the total number of sensitive sentences we filtered in sensitive dataset is denoted as m. We designed a pipeline comprising three steps: 1)Sensitive Sentence Identification, 2) Sensitive Dataset Construction, 3) Keyword Extraction. The time complexity of Sensitive Sentence Identification is $O(n \times L(s))$, Sensitive Dataset Construction is $O(n)$, and Keyword Extraction is $O(m)$. As the steps are executed sequentially, the overall time complexity of the pipeline is $O(n \times L(s) + n + m)$. Specifically, we utilized the NVIDIA GeForce RTX 3060 as our testing device. In each iteration, our queries totaled approximately 20,000, incurring a cost of merely about 0.07 GPU-days. This implies that the execution of our process is not significantly time-consuming.

We also analyzed the time taken with different numbers of queries. As shown in Table IX, an increase in the number of queries necessitates a longer duration to generate and identify sensitive sentences. However, even for 30,000 queries, the time consumption is only 0.13 GPU-days, which is not significantly high. During the extraction process, the time cost remains minimal, estimated in tens of milliseconds, with negligible differences observed across varying numbers of queries.

B. The Profound Privacy Implications

In our domain keyword inference attack, although we observe only over a 3% improvement (PubMed) and a 6% improvement (Rotten Tomatoes) in F1 score in our domain keyword inference attack, it is imperative to highlight the profound privacy implications that even such seemingly small proportions can have. Typically, training a language model with good generalization necessitates a large amount of data. Therefore, when this improvement is applied to large training datasets, it might lead to considerable leakage of training data. In our experiment, we have recovered a total of 3,335 keywords on which the target model is trained on PubMed, whereas for the baseline sample recovery, it only recovered a total of 1,791 keywords.

Moreover, although we only extract about 0.3% personal information in our private keyword inference attack, it is imperative to highlight the profound privacy implications even such a seemingly small proportion can have. For example, an adversary extracting sensitive personal information of passengers, including flight details, nationalities, and names, can expose individuals to multiple threats such as identity theft, phishing attacks, authentication issues, personal reputation harm, and potentially tracking and surveillance. Moreover, training a language model with good generalization typically necessitates a large amount of data. Therefore, when this proportion (0.3%) is applied to large training datasets, it might lead to considerable leakage of personal information. In our experiment, We extracted 128 correct pieces of personal information, accounting for 0.3% of the original training set, of which 40,000 pieces of personal information were used for training.

VIII. Conclusion and Future Work

In this paper, we propose a novel keyword inference attack framework to extract sensitive keywords by analyzing the fine-tuning leakage of the pre-train and then fine-tuning the paradigm. We first identify sensitive sentences using a carefully designed contrast difference score and the zlib entropy. We fine-tune the pre-trained LM to approximate the target model and maximally extract sensitive keywords. We implement two types of attacks according to our framework: domain and private keyword inference attacks. We also conduct comprehensive experiments on three types of downstream applications to empirically evaluate the performance of our attacks. We hope our attacks can attract the attention of researchers in the fine-tuning leakage and heat the arms race between privacy attacks and defenses. Our future work is to extend our current threat model structure to encompass a broader range of attacks and find correspond defend strategies to mitigate these risks by meticulously considering the data utilized for fine-tuning, implementing privacy-preserving techniques, and adhere to best practices for model deployment.

Acknowledgment

The authors thank the anonymous reviewers for their valuable comments.

References

[1] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, and I. Androutsopoulos, "Large-Scale multi-label text classification on EU legislation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 6314–6322.

[2] A. Resende, D. Railsback, R. Dowsley, A. C. A. Nascimento, and D. F. Aranha, "Fast privacy-preserving text classification based on secure multiparty computation," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 428–442, 2022.

[3] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ questions for machine comprehension of text," in *Proc. Empirical Methods Natural Lang. Process.*, 2016, pp. 2383–2392.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[6] T. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[7] Y. Sun et al., "Ernie 2.0: A continual pre-training framework for language understanding," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 8968–8975.

[8] Y. Zhang et al., "DIALOGPT: Large-scale generative pre-training for conversational response generation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2020, pp. 270–278.

[9] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 196–206.

[10] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *Proc. IEEE 31st Comput. Secur. Found. Symp. (CSF)*, Jul. 2018, pp. 268–282.

[11] S. Zanella-Béguelin et al., "Analyzing information leakage of updates to natural language models," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 363–375.

[12] N. Carlini et al., "Extracting training data from large language models," in *Proc. 30th USENIX Secur. Symp.*, 2021, pp. 2633–2650.
[13] S. Guo, C. Xie, J. Li, L. Lyu, and T. Zhang, "Threats to pre-trained language models: Survey and taxonomy," 2022, *arXiv:2202.06862*.
[14] C. Song and A. Raghunathan, "Information leakage in embedding models," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2020, pp. 377–390.
[15] F. Mireshghallah, A. Uniyal, T. Wang, D. Evans, and T. Berg-Kirkpatrick, "Memorization in NLP fine-tuning methods," 2022, *arXiv:2205.12506*.
[16] R. Panchendrarajan and S. Bhoi, "Dataset reconstruction attack against language models," in *Proc. CEUR Workshop*, 2021, pp. 1–17.
[17] H. A. Inan et al., "Training data leakage analysis in language models," 2021, *arXiv:2101.05405*.
[18] T. Vakili and H. Dalianis, "Are clinical BERT models privacy preserving? The difficulty of extracting patient-condition associations," in *Proc. HUMAN@ AAAI Fall Symp.*, 2021.
[19] E. Lehman, S. Jain, K. Pichotta, Y. Goldberg, and B. Wallace, "Does BERT pretrained on clinical notes reveal sensitive data?" in *Proc. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2021, pp. 946–959.
[20] J. Huang, H. Shao, and K. Chen-Chuan Chang, "Are large pre-trained language models leaking your personal information?" 2022, *arXiv:2205.12628*.
[21] P. Regulation, "General data protection regulation," in *Proc. Intouch*, vol. 25, 2018, pp. 1–5.
[22] Q. Wang, P. Liu, Z. Zhu, H. Yin, Q. Zhang, and L. Zhang, "A text abstraction summary model based on BERT word embedding and reinforcement learning," *Appl. Sci.*, vol. 9, no. 21, p. 4701, Nov. 2019.
[23] X. Pan, M. Zhang, S. Ji, and M. Yang, "Privacy risks of general-purpose language models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 1314–1331.
[24] Q. Feng, D. He, Z. Liu, H. Wang, and K. R. Choo, "SecureNLP: A system for multi-party privacy-preserving natural language processing," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3709–3721, 2020.
[25] G. L. Prajapati and R. Saha, "REEDS: Relevance and enhanced entropy based Dempster Shafer approach for next word prediction using language model," *J. Comput. Sci.*, vol. 35, pp. 1–11, Jul. 2019.
[26] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.
[27] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," OpenAI, 2018.
[28] U. Germann, "Greedy decoding for statistical machine translation in almost linear time," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2003, pp. 72–79.
[29] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," 2018, *arXiv:1805.04833*.
[30] C. Tillmann and H. Ney, "Word reordering and a dynamic programming beam search algorithm for statistical machine translation," *Comput. Linguistics*, vol. 29, no. 1, pp. 97–133, Mar. 2003.
[31] S. Yilmaz, E. Aydogan, and S. Sen, "A transfer learning approach for securing resource-constrained IoT devices," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4405–4418, 2021.
[32] P. Deutsch, "RFC1952: GZIP file format specification version 4.3," RFC Editor, USA, 1996.
[33] P. Deutsch, *GZIP File Format Specification Version 4.3*, RFC, 1996. [Online]. Available: https://dl.acm.org/doi/book/10.17487/RFC1952
[34] N. Giarelis, N. Kanakaris, and N. Karacapilidis, "A comparative assessment of state-of-the-art methods for multilingual unsupervised keyphrase extraction," in *Proc. Artif. Intell. Appl. Innov., IFIP WG 12.5 Int. Conf.*, 2021, pp. 635–645.
[35] E. Alsentzer et al., "Publicly available clinical BERT embeddings," 2019, *arXiv:1904.03323*.
[36] Z. Liu, F. Jiang, Y. Hu, C. Shi, and P. Fung, "NER-BERT: A pre-trained model for low-resource entity tagging," 2021, *arXiv:2112.00405*.
[37] H. B. Barlow, "Unsupervised learning," *Neural Comput.*, vol. 1, no. 3, pp. 295–311, 1989.
[38] M. Abadi et al., "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.
[39] Z. Lu, H. J. Asghar, M. A. Kaafar, D. Webb, and P. Dickinson, "A differentially private framework for deep learning with convexified loss functions," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 2151–2165, 2022.
[40] J. M. Joyce, "Kullback–Leibler divergence," in *Proc. Int. Encyclopedia Stat. Sci.*, 2011, pp. 720–722.
[41] F. Dernoncourt and J. Young Lee, "PubMed 200k RCT: A dataset for sequential sentence classification in medical abstracts," 2017, *arXiv:1710.06071*.
[42] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," 2015, *arXiv:1508.07909*.
[43] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, pp. 1–13, Dec. 2020.
[44] J. Chu, Y. Liu, Z. Yang, X. Shen, M. Backes, and Y. Zhang, "Comprehensive assessment of jailbreak attacks against LLMs," 2024, *arXiv:2402.05668*.
[45] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, "'Do anything now': Characterizing and evaluating in-the-wild jailbreak prompts on large language models," 2023, *arXiv:2308.03825*.
[46] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How does LLM safety training fail?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 1–32.
[47] Z.-X. Yong, C. Menghini, and S. H. Bach, "Low-resource languages jailbreak GPT-4," 2023, *arXiv:2310.02446*.
[48] W. Li and J. Zhao, "TextRank algorithm by exploiting Wikipedia for short text keywords extraction," in *Proc. 3rd Int. Conf. Inf. Sci. Control Eng. (ICISCE)*, Jul. 2016, pp. 683–686.
[49] M. Bianchini, M. Gori, and F. Scarselli, "Inside PageRank," *ACM Trans. Internet Technol.*, vol. 5, no. 1, pp. 92–128, Feb. 2005.
[50] J. Ramos et al., "Using tf-idf to determine word relevance in document queries," in *Proc. 1st Instructional Conf. Mach. Learn.*, 2003, pp. 29–48.
[51] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "YAKE! Keyword extraction from single documents using multiple local features," *Inf. Sci.*, vol. 509, pp. 257–289, Jan. 2020.
[52] N. Hardeniya, J. Perkins, D. Chopra, N. Joshi, and I. Mathur, *Natural Language Processing: Python and NLTK*. Birmingham, U.K.: Packt Publishing, 2016.
[53] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.
[54] C. D. Manning et al., "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2014, pp. 55–60.

Zhirui Zeng received the B.S. degree from Chongqing University in 2021, where he is currently pursuing the master's degree with the College of Computer Science. His research focuses on computer system security. He is particularly interested in security threats and attacks in machine learning systems and natural language process systems.

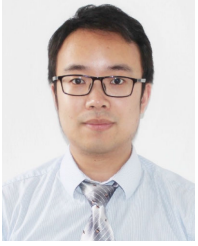
Tao Xiang (Senior Member, IEEE) received the B.Eng., M.S., and Ph.D. degrees in computer science from Chongqing University, China, in 2003, 2005, and 2008, respectively. He is currently a Professor with the College of Computer Science, Chongqing University. He has published over 100 papers on international journals and conferences. His research interests include multimedia security, cloud security, data privacy, and cryptography. He served as a referee for numerous international journals and conferences.

Shangwei Guo (Member, IEEE) received the Ph.D. degree in computer science from Chongqing University, Chongqing, China, in 2017. He is an Associate Professor with the College of Computer Science, Chongqing University. He was a Post-Doctoral Research Fellow with Hong Kong Baptist University and Nanyang Technological University, from 2018 to 2020. His research interests include machine learning security and multimedia security.

Jialing He (Member, IEEE) received the M.S. and Ph.D. degrees from Beijing Institute of Technology, Beijing, China, in 2018 and 2022, respectively. She is currently a Research Assistant Professor with the College of Computer Science, Chongqing University, Chongqing, China. Her current research interests include machine learning security, differential privacy, and blockchain.

Qiao Zhang (Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, Old Dominion University, in 2021. She is an Assistant Professor with the College of Computer Science, Chongqing University, China. Her current research is about the privacy-preserved machine learning. She has published papers, such as in NDSS, AsiaCCS, IJCAI, *IEEE Network*, and IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING.

Guowen Xu (Member, IEEE) received the Ph.D. degree in cyberspace security from the University of Electronic Science and Technology of China (UESTC) in 2020. He is currently a Research Fellow with Nanyang Technological University, Singapore. As the first author, he has published more than 20 papers in top international conferences and journals, including ACM CCS, ACM ACSAC, ACM ASIACCS, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, and IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY. His research interests include secure outsourcing computing and privacy-preserving issues in deep learning. He was a recipient of the Best Paper Award of the 26th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2020) and the IEEE INFOCOM Student Conference Award.



Tianwei Zhang (Member, IEEE) received the bachelor's degree from Peking University in 2011, and the Ph.D. degree from Princeton University in 2017. He is an Assistant Professor with the School of Computer Science and Engineering, Nanyang Technological University. His research focuses on computer system security. He is particularly interested in security threats and defenses in machine learning systems, autonomous systems, computer architecture, and distributed systems.