ulm university universität
uulm

Ulm University | 89069 Ulm | Germany

Faculty for Engineering, Computer Science and Psychology
Institute of Databases and Information Systems (DBIS)

Does BERT Memorize Named Entities? Extracting Training Data from Fine-tuned BERT models

Master's Thesis at Ulm University

Presented by:
Andor Diera
andor.diera@uni-ulm.de
1052652

Supervisors:
Prof. Dr.-Ing. Ansgar Scherp
Prof. Dr. rer. nat. Frank Kargl
Nicolas Lell

Industrial Supervisors:
Aygul Garifullina (aygul.garifullina@bt.com)

2022

Version November 15, 2022

Name: Andor Diera Student Number: 1052652

Statement of Originality

I hereby declare that I have written the thesis by myself, without contributions from any sources or aids other than those indicated. I confirm that this work has not been submitted or published elsewhere in any other form for the fulfillment of any other degree or qualification.

. . . . . .
Place and Date

. . . . . .
Andor Diera

Contents

List of Figures

List of Tables

LIST OF ACRONYMS

DNN - Deep Neural Network

DP - Differential Privacy

DPSGD - Differentially Private Stochastic Gradient Descent

GAN - General Adversarial Network

GPT - General Pre-trained Transformer

LM - Language Model

ML - Machine Learning

MLM - Masked Language Modeling

NER - Named Entity Recognition

NLG - Natural Language Generation

NLP - Natural Language Processing

NLU - Natural Language Understanding

NSP - Next Sentence Prediction

PATE - Private Aggregation of Teacher Ensembles

PII - Personally Identifiable Information

PPDL - Privacy Preserving Deep Learning

VAE - Variational Autoencoder

Andor Diera
andor.diera@uni-ulm.de
Ulm University

ABSTRACT

The large-scale data collection required for deep learning introduces numerous privacy issues. Privacy preserving deep learning is an emerging field in machine learning that aims to mitigate the privacy risks in the use of deep neural networks. One such risk is training data extraction from language models that have been trained on datasets that contain personal and privacy sensitive information. In our study we investigate the extent of named entity memorization in fine-tuned BERT models. We use single-label text classification as a downstream task and employ three different fine-tuning setup in our experiments, including one with Differentially Privacy. We create a large number of text samples from the fine-tuned BERT models utilizing a custom sequential sampling strategy with two prompting strategies, and search in these samples for named entities also present in the fine-tuning datasets. The results show that our models do not generate more fine-tuning specific named entities entities than an only pre-trained model that has not been trained on the fine-tuning datasets. This suggests that it is highly unlikely to extract personal or privacy sensitive named entities from fine-tuned BERT models. Furthermore, we show that the application of Differential Privacy has a huge effect on the text generation capabilities of BERT, making it a promising privacy preserving method for other language models that are more prone to training data extraction attacks.

KEYWORDS

privacy preserving deep learning, language models, training data extraction, named entities, text generation, differential privacy

1 INTRODUCTION

The field of Machine Learning (ML) has seen an enormous growth in the past decade and revolutionized a wide variety of application domains. One of the main improvements in the area was the emergence of Deep Neural Networks (DNNs). With the rise of computational power and the increase in the amount of data available to researchers, DNNs became the forerunners of the recent successes in many ML domains, such as image recognition and natural language processing (NLP) [31]. Although utilizing large volumes of training data is one of the main driving factors behind the great performance of modern ML models, publishing these models to the public raises some serious privacy concerns regarding private and confidential information present in the training data [35]. These privacy concerns are especially relevant for large Language Models (LMs) which form the basis of most state-of-the-art technologies in a large range of NLP tasks [5, 6].

LMs are defined as statistical models which assign a probability to a sequence of words. Recent versions of these models are usually first pre-trained in a task-agnostic self-supervised manner. The latest large LMs use a corpus size ranging from hundreds of gigabytes to several terabytes of text [4, 49] during this self-supervised process. The sheer size of these datasets makes it near impossible for researchers to remove all confidential information which may be present in the corpus and a recent study has shown that it is possible to extract personal information from some large LMs, even if that given information has only appeared once in the training corpus [6].

While the training cost of these large LMs became so prohibitively expensive that only the biggest tech companies can afford it [55], pre-trained LMs are commonly used in businesses that work with huge amounts of text data. These businesses include banks, telecommunication, and insurance companies, which often handle a great amount of personal and privacy sensitive data. In practice, pre-trained LMs are fine-tuned on a business-specific dataset using some downstream task (such as text-classification, question-answering or natural language inference) before deployment [10]. Although the fine-tuning may mitigate some of the unintended memorization of the original dataset used in pre-training, it raises new concerns regarding the personal and privacy sensitive information in the business-specific dataset used during the fine-tuning process [6].

Privacy Preserving Deep Learning (PPDL) is a common term used for methods aiming to mitigate general privacy concerns present in the use of DNNs. Multiple approaches have been proposed to achieve PPDL [42], but so far there is no perfect solution to this problem, with each method having its own challenges and limitations. The most popular techniques include Federated Learning [40], the application of Differential Privacy (DP) [1, 73], encryption [3, 23], and data anonymization [57].

In this study we investigate whether it is possible to extract personal information from one of the most popular modern LMs, BERT [10]. BERT is an Auto-Encoder Transformer that is mostly used for Natural Language Understanding (NGU) tasks. Since BERT is less adept at generating long, coherent text sequences [62], we focus our study on the generation and extraction of named entities. We conduct our experiments on fine-tuned models to better understand the privacy risks involved in using BERT for commercial purposes. In total, we investigate three different fine-tuning setups - including one with the application of DP, on two different datasets. We assess each configuration with regard to the performance on single-label text classification and to the extent of named entity memorization. In summary the main contributions of the paper are:

- The proposal of a framework to study the memorization of named entities in fine-tuned LMs.
- The evaluation of named entity extraction methods on fine-tuned and only pre-trained BERT models.
- The assessment of the effects of Differentially Private fine-tuning on model performance and privacy.

The subsequent section discusses the related work in the field. It serves as the foundation for defining our framework and methods for extracting named entities from fine-tuned BERT models, as described in Section 3. Section 4 describes the datasets and the implementation details of the experiments. The results are described in Section 5 and discussed in Section 6, before we conclude in Section 7.

2 RELATED WORK

In this section, we briefly review the literature on LMs with a focus on the BERT model and text generation, possible privacy attacks against DNNs and existing approaches to PPDL.

2.1 Language Models

Modern large LMs rely on two core concepts that led to their dominance in the NLP field: the focus on the self-attention mechanism in the DNN architecture and the introduction of large scale task-agnostic pre-training to learn general language representation [65].

Self-attention is used for modeling dependencies between different parts of a sequence. A landmark study in 2017 [60] has shown that self-attention was the single most important part of the state-of-the-art NLP models of that time, and introduced a new family of models called Transformers, which rely solely on stacked layers of self-attention and feed-forward layers. Besides the state-of-the-art performances, another great advantage of the Transformer architecture is that unlike a recurrent architecture, it allows for training parallelization.

The ability to parallelize training, alongside the significant increase in computational power allowed these models to train on larger datasets than once was possible. Since supervised training requires labeled data, self-supervised pre-training with supervised fine-tuning became the standard approach when using these models [39]. The first Transformer that achieved great success on a large-range of NLP tasks using this approach was the General Pre-trained Transformer (GPT) [48].

State-of-the-art Transformers can be divided in to three main categories based on their pre-training approach: Auto-Regressive models which use the classical language modelling pre-training task of next word prediction, Auto-Encoding models which are pre-trained by reconstructing sentences that have been corrupted in some way, and Seq2seq models which usually employ objectives of encoding-decoding models for pre-training [70].

BERT. A major limitation of the Auto-Regressive models is that during pre-training they learn a unidirectional language model. In these models tokens are restricted to only attend other tokens left to them, therefore they can only predict a token based on the context from the left. In one of the first Auto-Encoding Transformer model a bidirectional approach has been proposed in the paper titled "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" [10]. This model utilizes Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) as pre-training tasks. In MLM a token is randomly removed from the input sequence and the model is trained to predict the removed token using context from both directions. Formally the objective is to learn the probability distribution over vocabulary V for masked token x_{mask} in sentence X such that:

$$\mathrm{x}_{mask} = \underset{w \in \mathcal{V}}{\operatorname{argmax}} \Pr(\mathrm{w}|\mathrm{X}\backslash\mathrm{x}_{mask})$$

Although the parameter count of BERT is greatly surpassed by more recent large LMs (such as GPT-3), BERT is still one of the most common baseline in many NLP benchmark tasks. The strong performance coupled with the fact that the model is democratized and has publicly available pre-trained implementations makes BERT a popular choice of NLP model both in industry and academia. Since its original release there have been dozens of follow-up studies and models published [51]. The most notable variants include DistilBERT - a general purpose distilled model of the pre-trained BERT [53], RoBERTa - an improved version of BERT which utilizes advanced hyperparameter optimization and a modified MLM method on a larger pre-training corpus [37] and ALBERT a "lite" version of BERT using factorized embedding parameterization, cross-layer parameter sharing and sentence-order prediction instead of NSP during pre-training [30].

Natural Language Generation. Natural Language Generation (NLG) is a subfield of NLP that is focused on the process of producing natural language text that enables computers to write like humans [70]. It lies at the core of many applied NLP domains, such as Machine Translation [71], Text Summarization [44] and Dialoge Systems [7]. Early approaches in text generation used Recurrent Neural Networks [15] and deep generative models, most notably Variational Autoencoders [28] to varying degree of success, but the introduction of large pre-trained Transformer pushed the state-of-the-art in NLG as well.

Although Auto-Regressive Transformer models are the standard choice for the task of NLG (since they are already trained to predict the next token based solely on previous tokens in a sequence), it has been shown that BERT can also be utilized to generate reasonably coherent text. Wang and Cho [62] designed a generation strategy for BERT based on Gibbs sampling [21], where given a seed sequence, tokens at random positions are masked and replaced by new tokens based on the sampling technique. Another generation strategy developed for Auto-Encoding Transformers [22] used a method of taking a fully masked sequence, predicting all tokens at once and then iteratively re-masking the token with the lowest probability and replacing it with a newly computed token.

2.2 Privacy Attacks in Machine Learning

Privacy attacks in ML denote a specific type of adversarial attack, which aim to extract information from a ML model. Based on recent surveys in the field [9, 35, 50], these attacks can be divided into five main categories.

Membership Inference Attacks. The goal of a membership inference attack is to determine whether or not an individual data instance is part of the training dataset for a given model. This attack typically assumes a black-box query access to the model. The common approach to this type of attack is to use a shadow training technique to imitate the behavior of a specific target model. The trained inference model is then used to recognize differences on the target model predictions between inputs used for training and inputs not present in the training data [56]. Membership inference attacks have been shown to work on models used for supervised classification tasks [56] , Generative Adversarial Networks (GANs) [36], Variational Autoencoders(VAEs) [36], and the embedding layers of LMs [38].

Model Extraction Attacks. The adversarial aim of a model extraction attack is to duplicate (i.e., "steal") a given ML model. It achieves this by approximating a function f' that is the same as the function f of the target model [35]. A shadow training scheme has been shown to successfully extract popular ML models such as logistic regression, decision trees, and neural networks, using only a black-box query access [59]. Other works have proposed methods to extract information about hyperparameters [63] and properties of the architecture [45] in neural networks.

Model Inversion Attacks. The idea behind model inversion attacks states that an adversary can infer sensitive information about the input data using a target model's output. These attacks can be used to extract input features and/or reconstruct prototypes of a class in case the inferred feature characterize an entire class [9]. The first model inversion attack [18] was based on the assumption

that the adversary has white-box access to a linear regression model, with some prior knowledge about the features of the training data. With the use of the output labels and known values of non-sensitive features, this attack is capable of estimating values of a sensitive feature. Existing work was later extended to neural networks with a new type of model inversion attack [17], which reformulated the attack as an optimization problem where the objective function is based on the target model output and uses gradient descent in the input space to recover the input data point. This technique allows the adversary to reconstruct class prototypes (i.e., faces in a facial recognition model) given a white-box access to the model and knowledge about the target labels with some auxiliary information of the training data.

Property Inference Attacks. The goal of property inference attacks is to infer some hidden property of a training dataset that the owner of the target model does not intend to share (such as feature distribution or training bias). Initially, property inference attacks were applied on discriminative models with white-box access [41, 47]. A more recent work has extended the method to work on generative models with black-box access [47].

Training Data Extraction Attacks. Training data extraction attacks aim to reconstruct training datapoints, but unlike model inversion attacks, the goal is to retrieve verbatim training examples and not just "fuzzy" class representatives [6]. These attacks are best suited for generative sequence models such as LMs. Initially these attacks have been designed for small LMs using academic datasets [5, 58, 69]. The aim of these studies was to measure the presence of training datapoints when generating sequences that are irrelevant to the learning task and are unhelpful to improving model performance. A common approach to measure the extent of this unintended memorization is to insert so-called "canaries" (artificial datapoints) into the training datasets and quantify their occurrence during sequence completion [5]. Since these initial studies were based on smaller models trained with a high number of epochs, it was assumed that this kind of privacy leakage must be correlated with overfitting [69]. However, a follow-up study using the GPT-2 model (which is trained on a very large corpus for only a few epochs) showed that even state-of-the-art large LMs are susceptible to these kinds of attacks. Using the pre-trained GPT-2 model, Carlini et al. [6] were able to generate and select sequence samples which contained low *k-eidetic* data-points (data points that have a frequency of k in the training corpus). A later study on Clinical BERT [33] (a BERT variant pre-trained on clinical texts) attempted to extract patient-condition association using both domain specific template infilling and the text generation methods inspired by the text extraction research done on GPT-2 [6] and the BERT specific text generation technique proposed by Wang and Cho [62]. Their methods were not successful in reliably extracting privacy sensitive information from Clinical BERT, but it remains inconclusive whether its due to the limitations in their method or in the linguistic capabilities of BERT.

2.3 Privacy Preserving Deep Learning

Since deep learning is a subfield of ML, most methods developed for privacy preserving ML can be also adapted to PPDL. Based on the literature [9, 35, 50] these methods can be divided into five main categories.

Encryption. Cryptography-based methods can be divided into two subcategories, depending whether the target of the encryption is the training data [23] or the model [3]. Regardless of the target, most existing approaches use homomorphic encryption, which is a special kind of encryption scheme that allows computations to be performed on encrypted data without decrypting it in advance [2]. Since training a DNN is already computationally expensive, adding homomorphic encryption to the process raises major challenges as it increases training times by at least an order of magnitude [35].

Data Anonymization. Data Anonymization techniques aim to remove all Personally Identifiable Information (PII) from a dataset. The common approach to achieve this is to remove attributes that are identifiers and mask quasi-identifier attributes [66]. The popular k-anonymity algorithm [57] works by suppressing identifiers (i.e., replacing them with an asterisk) and generalizing quasi-identifiers with a broader category which has a frequency of at least k in the dataset. Although data anonymization techniques were developed for structured data, it is possible to adapt them to unstructured text data as well [24].

Differentially Private Learning. Differential Privacy (DP) is a rigorous mathematical definition of privacy in the context of statistical and machine learning analysis. It addresses the paradox of "learning nothing about an individual while learning useful information about the population" [14]. In ML, DP algorithms aim to obfuscate either the training data [72] or the model [52], by adding noise. Since DNN parameters are highly sensitive to noise, the best place for applying DP in deep learning is the gradients. Abadi et al. [1] proposed an efficient training algorithm with a modest privacy budget called Differentially Private Stochastic Gradient Descent (DPSGD). DPSGD ensures DP by cutting the gradients to a maximum l2 norm for each layer and then adding noise to the gradients bounded by the "l2 norm-clipping-bound". Although DPSGD comes with increased computational cost and performance loss, variations of this algorithm [8, 12] still belong to the cutting-edge of PPDL research.

Aggregation. Aggregation methods are generally used along with distributed/collaborated learning, in which multiple parties join a ML task while aiming to keep their respective datasets private [35]. The most popular collaborative framework for privacy preservation is Federated Learning introduced by Google [40]. Although aggregation methods can provide data security during distributed training, their privacy preserving aspects are more limited than other PPDL approaches.

Combined Approaches. The four main categories of PPDL methods are not mutually exclusive. DP is often used in collaborated learning where it is combined with aggregation techniques [64]. A promising framework called Private Aggregation of Teacher Ensembles (PATE) [46] proposes improved privacy preservation with the use of an ensemble of teacher models (which have been trained on non-overlapping datasets), and a differentially private aggregation mechanism. The knowledge of the aggregated model is then transferred into a student model, resulting in a model with strong privacy guarantees.

2.4 Summary

Transformer-based architectures pushed the state-of-the-art in most areas of NLP. The BERT model [10] is still widely applied in most NLP tasks and even though it is not commonly used for

text generation due to its bidirectional pre-training approach, it can be still utilized to generate sentences of low linguistic quality [62]. It is possible to extract various type of information from ML models using privacy attacks [9, 35, 50]. Large LMs are prone to training data extraction attacks, a study on GPT-2 has shown that it is possible to extract privacy sensitive information from the training data with black-box access, using different prompting techniques when generating text [5]. As privacy attacks become more frequent against DNNs, the role of privacy preservation in deep learning also becomes more crucial. Although there are multiple ways to improve the privacy of a model including the use Differential Privacy during training [1], there is no perfect solution as of yet [9, 35, 50].

3 EXTRACTING MEMORIZED NAMED ENTITIES FROM BERT

In order to extract the named entities of the fine-tuning dataset from the BERT model, we present the experimental pipeline depicted in Figure 1. The pipeline consists of three main phases: the fine-tuning (including a privacy preserving approach using Differential Privacy), the text generation from the fine-tuned models, and the evaluation of the named entity memorization.

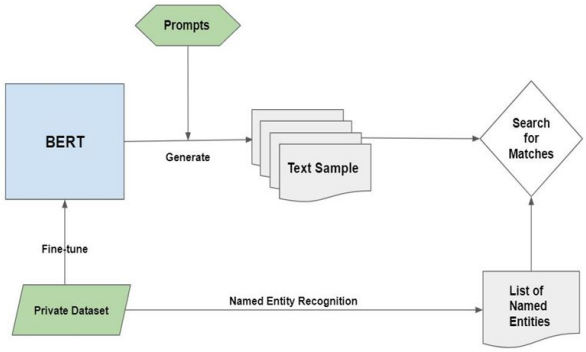

Figure 1: An illustration for our framework of extracting training from BERT. First we fine-tune a pre-trained BERT on a private dataset. Next we generate text samples from the fine-tuned model. Finally we search the generated samples for the named entities that were prior extracted from the private dataset.

3.1 Fine-tuning

In the fine-tuning phase, we employ single-label text-classification as the downstream task. Our setup consists of three different fine-tuning methods:

- Full
- Partial
- Differentially Private (DP)

The *Full* setup follows the standard practices of fine-tuning LMs, where a classifier head is attached to the base network and all the weights of a pre-trained network along with the classifier head are retrained on the task-specific dataset with a low learning rate [27]. Full fine-tuning usually leads to the best results on the downstream task, but in the case of large LMs, it is not always feasible due to the size of these networks and the computational costs of retraining them. Due to this constraint, researchers have designed alternative fine-tuning strategies where fine-tuning is employed in a more optimized manner [27, 32]. A common alternative strategy is to freeze most of the layers in a network and only retrain the last few encoder layers with task-specific head of the network. In our *Partial* setup we freeze all layers of the BERT model except for the last encoding layer. Applying DP in fine-tuning puts additional noise to the gradient updates, which in the lower layers carries a detrimental effect to the pre-trained knowledge of the model as the weights of the bottom layers are more sensitive to noise. For this reason, in the *Differentially Private* setup we employ the same layer freezing approach as in the *Partial* setup. The different fine-tuning methods are depicted in Figure 2.

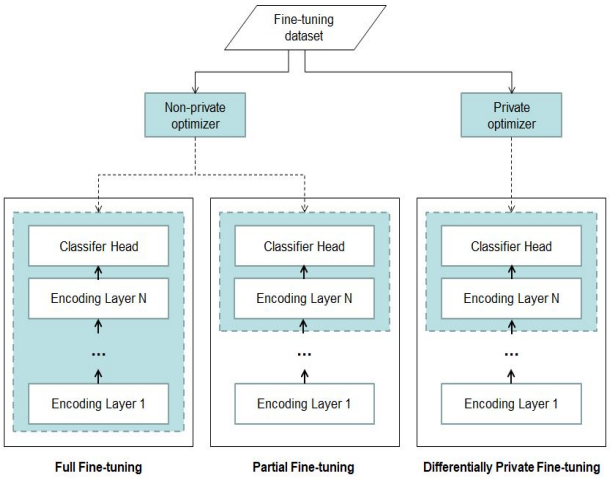

Figure 2: An illustration of the difference in the fine-tuning methods. In the *Full* setup we apply standard fine-tuning for all layers. In the *Partial* setup we only fine-tune the last encoding layer and the classifier head. For the *Differentially Private* we also employ partial fine-tuning but use a private-optimizer (DPSGD).

Differentially Private Stochastic Gradient Descent. In the *DP* setup we use the Adam variant of DPSGD for fine-tuning outlined by Abadi et al. [1]. DPSGD is a modification to the stochastic gradient descent algorithm that employs (ϵ, δ) differential privacy [13]. In the formal definition of (ϵ, δ) differential privacy, a randomized algorithm M is differentially private if:

$$\Pr[M(x) \in S] \leq \exp(\epsilon)\Pr[M(y) \in S] + \delta$$

Where S denotes all the potential output of M that can be predicted, ϵ is the metric of privacy loss (also known as the privacy budget) at a differential change in the data (e.g., adding or removing one datapoint) and δ is the probability of an accidental privacy leak. In deep learning, an ϵ value is defined as modest when it is below 10 [1] and δ is usually set to the the reciprocal of the number of training samples [12, 68].

The DPSGD algorithm entails two major changes to the gradient descent algorithm: the introduction of gradient clipping and the addition of Gaussian noise to the clipped gradients. The clipping ensures that each training point has a limit on how much it

[1] In standard data analytics settings, ϵ values between 0 and 1 are considered to be highly private, and values between 2 and 10 are considered somewhat private. However, in deep learning it is hard to achieve a ϵ value under 1 due to continuous increase of ϵ between training epochs.

can impact the model parameters, while the addition of the noise randomizes the behaviour of the algorithm, making it statistically impossible to know whether or not a training point was included in the training set. These modifications to the gradients happen on a microbatch level (ideally for each individual training sample) and are aggregated for the standard minibatch optimization step.

3.2 Text Generation

Although the standard objective of NLG is to produce text that appears indistinguishable from human-written text, in our study we are less interested in general text quality. Our primary goal is to force the fine-tuned models to generate named entities found in the training data. To achieve this, we employ two different prompting methods and an efficient generation strategy that produces diverse text samples.

Prompt Selection. Selecting good prompts is a crucial step in forcing the model to unveil information about the training data. We employ two different prompting strategies for text generation. In the first one we take the strategy shown to achieve the best results in the experiments of Carlini et al. [6], in which a fixed length of string sequence is randomly sampled as prompt from the Crommon Crawl[2] dataset. This we refer to as a *naive prompting*, since we randomly use text samples scraped from the internet. In the second strategy, we create the prompts by randomly selecting string sequences of a fixed length from the test set of the fine-tuning data. This setup is considered as *informed prompting* given that the prompts come from the same domain and are generally highly similar to the training data.

Text Generation. Despite the fact that the bidirectional nature of BERT does not naturally admit to sequential sampling, Wang and Cho [62] has shown that it is also possible to utilize this strategy for BERT. Although their results suggest that their non-sequential iterative method produces slightly more coherent text than sequential sampling, it requires multiple iterations for each token. Since text coherence is not our primary goal, we choose to employ a computationally less expensive sequential sampling method. In this method, we choose a randomly selected prompt as a seed sequence and extend it with a masked token. For each timestep we predict the masked token and then add an additional masked token to the extended sequence until we reach the defined sequence length.

On top of this generation method, we also employ a combination of beam search and nucleus sampling as an additional decoding strategy. The beam search algorithm is commonly used decoding method in machine translation tasks [19]. Compared to greedy search, where a at each timestep only the token with the highest probability is selected, beam search selects multiple tokens (defined by the beam size parameter) each timestep, and uses conditional probability to construct the best combination of these tokens in a sequence. While both greedy search and beam search select tokens based on maximum likelihood, sampling from the probability distribution is also a viable method. The most popular sampling method is top-k sampling, where in each timestep a token is sampled from a set of k candidates with the highest probability. Nucleus sampling (also called top-p sampling) is an alternative strategy to top k, where instead of using a set with a fixed length, the smallest possible set is constructed with tokens whose cumulative probability exceeds the probability value of parameter p [25]. These sampling methods can be combined with both search algorithms.

[2]http://commoncrawl.org

3.3 Evaluating Named Entity Memorization

Named entities refer to objects and instances that identify one item from a set of other items sharing similar attributes. They usually include names, organizations, locations, products, and special temporal or numerical expressions like dates or amounts of money [43]. In order to evaluate the extent to which the models have memorized the named entities found in the fine-tuning dataset, we first extract them from the datasets using Named Entity Recognition (NER) and create a dictionary with the entities and their corresponding label types. After this we create three different entity sets from this dictionary. The first one (*All*) consists of every entity with a character length greater than 3. In the second (*Private*), we do a cross-check of the first entity set with the pre-training data, and remove all entities also present in the pre-training datasets, leaving us a set of entities that only appear in the fine-tuning data. For the third and final set (*Private 1-eidetic*), we keep all 1-eidetic entities (i.e., entities which appeared only once in the fine-tuning dataset) of second set and discard everything else. Once we have these three sets and the generated samples from each model setup (including the samples generated from a base model that was not fine-tuned), we look for the number of exact matches in the samples.

4 EXPERIMENTAL APPARATUS

4.1 Datasets

For selecting suitable datasets for the study we had two criteria. First, to avoid privacy issues and ethical concerns only publicly available datasets were chosen. Second, to provide a good basis for the measurement of memorization, we were interested in datasets that contain a large number of named entities. We choose data which has English as its primary language and kept 20% of each dataset for testing. The label distribution of each dataset is shown in Appendix A, while the main characteristics of the datasets can be seen in Table 1.

Enron Email Dataset.

General Description. The raw Enron Email corpus [29] consists of 619,446 email messages from 158 employees of the Enron corporation, made public due to legal investigations. The cleaned version has 200,399 messages and is commonly used for various NLP tasks. Since this dataset contains full emails of real users, it includes naturally occurring personal information (such as names, addresses, organizations, social security numbers etc), which makes it a perfect fit for our research.

Preprocessing. Since the original dataset is not fitted to text-classification (as it lacks any official labels), we adapt it by labeling the emails by the folders names they are attached to (i.e. "sent-mail", "corporate" , "junk", "proposals" etc.). In total there are 1,781 folders in the dataset but most of the folders contain less than a couple of emails. Therefore, we first selected the folders with more than a 100 emails, and hand-picked folders that could be considered as valid classes in an applied setting. During the preprocessing of the emails, we removed the forward blocks, line breaks, and tabs. A detailed breakdown of the preprocessing can be found in Appendix A.

Blog Authorship Corpus.

General Description. The Blog Authorship Corpus [54] contains text from blogs written in 2004 and before, with each blog being the work of a single user. The data was collected from blogger.com in 2004. The corpus incorporates a total of 681,288 posts from 19,320 users. Alongside the blogposts, the dataset includes a topic label and demographic information about the writer, including gender, age, and zodiac sign. Although the blogposts were written for the public, they contain some Personally Identifiable Information (PII), such as names, organizations and postal addresses. We adapt the dataset for text-classification by labeling the posts by the topics.

Preprocessing. Posts with the topic label "unknown" were removed. After the removal the dataset consists of 430,269 posts with 39 unique labels. Preprocessing on the blogposts was kept to a minimal: only non-printable ASCII characters (e.g., Korean letters) and URL links were removed, otherwise the text remained unchanged.

Table 1: Characteristics of the datasets

Dataset	N	#Train	#Test	#Classes
Enron	7,501	6,000	1,501	7
BlogAuthorship	430,269	344,215	86,054	39

4.2 Procedure & Implementation

Fine-tuning. The experiments are based on the Hugging Face implementation of the BERT base uncased model [65], which uses 12 layers of transformer blocks and is pre-trained on the English Wikipedia [16] and Book Corpus [74] datasets. For single-label classification, a custom classifier head is attached to the base model consisting of a Dropout and a Linear layer. In the *Full* and *Partial* setup we used the standard Adam optimizer, while in the *Differentially Private* fine-tuning we changed it to the DPAdam optimizer from the Opacus library [67] with a microbatch size of 1.

Text Generation. For text generation we first removed the classifier heads from the fine-tuned models and attached a pre-trained MLM head instead. We then used the sequential generation method described in Section 3, with the addition of beam search and nucleus sampling combined with a temperature parameter. During prompt creation, we sampled a 100 character length string either from the Common Crawl dataset (*naive prompting* or from the test set (*informed prompting*. We set the sequence length to 256 tokens and removed the prompt tokens before saving the samples. In total, we generated 20.000 samples for each setup.

Named Entity Recognition. For collecting the named entities from the fine-tuning datasets, we employed the NER system of the spaCy library that utilizes a custom word embedding strategy, a Transformer architecture, and a transition-based approach to named entity parsing [26]. spaCy distinguishes between a total of 18 different named entity types. Out of these 18 types we selected 7 (*Person, Organization, Location, Geo-Political Event, Facility, Money, Cardinal*) which have the highest possibility to contain personal or privacy sensitive information. A more detailed discussion about the type selection is found in Appendix B. When creating the three different sets of entities described in Section 3.3, we used the Book Corpus and Wikipedia datasets (available through the datasets library [34]) for cross-checking. The number of named entities in the three sets sorted by entity types can be seen in Table 2.

4.3 Hyperparameter Optimization

Fine-tuning. During fine-tuning, each configuration was carefully optimized for both datasets using manual tuning based on test accuracy. Dropout rates were fixed based on the default BERT base implementation of the huggingface library (0.1 for attention dropout and 0.3 for the classifier) [65]. For batch size, learning rate, and number of epochs a search space was defined based on previous works [11, 20]:

- batch size: { 8, 16, 32 }
- learning rate: { 5e-3, 1e-3, 1e-4, 5e-5, 1e-5 }
- number of epochs: { 3, 5, 10 }

In the *Full* configuration the best performing values were found to be the following:

- batch size: 32 (on both datasets)
- learning rate: 1e-5 (on both datasets)
- number of epochs: 10 (Enron), 5 (Blog Authorship)

In the *Partial* setup (when only a subset of the last layers got updated), The best results were found with the subsequent values:

- batch size: 32 (on both datasets)
- learning rate: 5e-5 (Enron), 1e-4 (Blog Authorship)
- number of epochs: 10 (Enron), 5 (Blog Authorship)

In the *DP* fine-tuning two additional hyperparameters had to be optimized to achieve the highest possible accuracy while keeping the privacy budget (ϵ) single-digit. We set the per-example gradient clipping threshold to 10 based on a previous study using DP with BERT [68], and found the best values for the noise multiplier to be 0.5 for the Enron and 0.4 for the Blog Authorship dataset. In conjunction with the DP optimization, we found the best values for the base hyperparameters to be the following:

- batch size: 32 (on both datasets)
- learning rate: 1e-3 (on both datasets)
- number of epochs: 10 (Enron), 5 (Blog Authorship)

Text Generation. During text-generation we studied the effects of the different sampling parameters and found the best results (in terms of text diversity and coherence) through manual tuning with the following value combinations:

- number of beams: 1
- beam size: 30
- nucleus sampling value: 0.8
- temperature: 2.0
- n-gram repetition limit: 3

4.4 Measures

Fine-tuning. To evaluate the performance on the downstream task we use the standard metric of single-label classification, namely accuracy (which is equivalent to Micro-F1 in single-label setups [20]). All experiments were conducted with the same random seed, training and inference time was also measured using a NVIDIA A100 HGX GPU with 40 GB of RAM.

Privacy Preservation. In the *DP* setup, we measure privacy preservation with the privacy budget ϵ. In all models the extent of unintended memorization of named entitites found in the fine-tuning dataset is measured by counting their occurrences in generated samples and checking their k-eidetic value.

Table 2: Number of named entities found in the datasets sorted by type. *All* refers to the entity set which contains every entity from the fine-tuning datasets, *Private* refers to the set of entities which are specific to the fine-tuning datasets, and *Private 1-eidetic* contains only the 1-eidetic entities from *Private* set.

Named Entity Type	Enron			Blog Authorship		
	All	Private	Private 1-eidetic	All	Private	Private 1-eidetic
PERSON	10,712	7,717	4,844	209,434	137,892	113,599
ORG	9,933	7,178	5,001	168,068	107,480	90,594
LOC	316	175	125	10,562	4,902	4,049
GPE	1,551	739	490	37,691	17,781	13,196
FAC	367	230	174	12,824	7,137	6,349
MONEY	1,220	736	585	11,216	7,551	6,343
CARDINAL	2,918	1,924	1,386	24,075	13,020	10,810
Total	27,017	18,726	12,605	473,870	295,763	244,940

5 RESULTS

Given the pipeline and configurations described in Section 3 and 4, we evaluate the results of the classification performance and the extent of unintended memorization of each model. We compare our findings on both datasets.

5.1 Classification

Table 3 shows the singe-label text classification results for each setup. In both datasets we observed a similar trend between the different fine-tuning setups: *Full* achieved the highest accuracy on the test set, *Partial* performed slightly worse, and the *DP* setup produced the worst results with a 15% drop compared to the *Partial*. In general, the accuracy values are considerably higher on the Enron dataset. In the *DP* setup the privacy budget ϵ is 9.79 for the Enron dataset and 7.38 for the Blog Authorship

Table 3: Accuracy scores after fine-tuning

Fine-tuning Setup	Enron	Blog Authorship
Full	87.5%	51.6%
Partial	85.8%	49.6%
DP	69.8%	35.7%

Runtime Performance of Fine-tuning. We provide the total running times in Table 4 as observed while conducting the experiments on a single NVIDIA A100-SXM4-40GB card. *Full* fine-tuning is the slowest due to the fact that optimization happens for all layers. When comparing the two setups when only the last layers are optimized, *DP* fine-tuning takes notably longer time than *Partial*, but is still faster than the *Full* setup.

Table 4: Train+inference runtime, rounded up to minutes

Fine-tuning Setup	Enron	Blog Authorship
Full	11	330
Partial	6	186
DP	9	222

5.2 Named Entity Memorization

For the named entity memorization experiments we also included a pre-trained BERT without any fine-tuning, which we called the *Base* setup. Figure 3 shows our initial results on the *All* entity set. The highest extraction rate was 9.3% for the Enron and 6.2% for the Blog Authorship dataset. On Enron the *Base* and *Partial* results were comparable, with a 0.2% percent difference in the naive prompting setup and a 0.6% percent in the informed prompting setup. On the Blog Authorship dataset a similar difference can be shown between the *Base* and *Full*, with the naive prompting having 0.5% and the informed prompting a 0.2% difference. The *DP* setup produced the lowest extraction rates with 1.4% and 1.1% on the Enron and 0.1% in the Blog Authorship dataset. Naive prompting consistently outperformed informed prompting in all fine-tuning setups on both datasets.

Figure 4 shows the comparison of the extraction rates between the private entity and the private 1-eidetic sets, using the naive prompting method. Compared to the results in Figure 3 the extraction rates are consistently lower across all setups. The difference between *Base* and *Partial* on Enron, and *Base* and *Full* on Blog Authorship is again negligible. Overall the memorization rate of private 1-eidetic entities is lower than the memorization rate of all private entities, but the difference is less than 1% on the Enron and less than 2% on the Blog Authorship dataset.

To further investigate the extracted private entities, we also recorded the extraction ratio of each entity type in Table 5. The Location and Geo-Political Event types produced the highest percentages, while the Facility and Money types had results less than 3% across all setups in both datasets. The extraction ratios on Blog Authorship are consistently lower in every entity type compared to Enron. The only exception can be seen in the Facility type, where the Blog Authorship results were 1-2% higher.

A more detailed result breakdown for each setup combination can be found in Appendix B.2.

6 DISCUSSION

6.1 Key Insights

Prompting Methods. Our experiments have shown that the naive prompting method produces better results in all setups. Although with the use of informed prompting the seed sequences will be more similar to the text sequences found in the fine-tuning data, it also limits the output diversity to some degree. Similarly to Carlini et al. [6], we argue that using random prompts sampled from a huge corpus unrelated to the training data yields better extraction results. This shows that adversaries do not need to have prior knowledge about the training data of the attacked model, a simple black-box approach can be sufficient.

Named Entity Memorization in BERT. Although we successfully extracted a small percentage of private named entities

Table 5: Extraction ratio of entities from the *Private* set using naive prompting, sorted by entity types

Named Entity Type	Enron				Blog Authorship			
	Base	Full	Partial	DP	Base	Full	Partial	DP
PERSON	4.1%	2.3%	4.3%	0.8%	3.4%	4.1%	2.9%	*
ORG	3.8%	2.2%	3.3%	0.3%	4.5%	4.1%	3%	*
LOC	20.5%	15.4%	18.4%	5.1%	8.9%	8.6%	5.5%	*
GPE	28.1%	22.5%	28%	5%	11.5%	13.4%	9.9%	0.1%
FAC	1.7%	0.4%	0.8%	0.8%	2.7%	2.5%	1.6%	*
MONEY	1.5%	0.7%	1%	0.1%	1.2%	0.9%	0.7%	*
CARDINAL	4.8%	1.7%	3.9%	0.5%	4.4%	3.9%	2.7%	0.1%

* less than 0.1%

from the fine-tuned models, our results show that using pre-trained model that has not been fine-tuned on the training set containing those entities, we can produce similar extraction ratios. Our assumption is that the low percent of private entities that have been successfully extracted from both the *Base* and *Full* or *Partial* models are low enough in complexity and length to be randomly generated from the combinations of common subword tokens. However, to fully understand the reasons behind this, a more detailed analysis of the extracted entities is needed. As can be seen from Table 5, distinct entity types have different probabilities to be extracted. From these 7 types, we argue Location and Geo-Political Event are the least unique in their nature, therefore it is not suprising that the highest extraction rates have been achieved on them. The lower values in the Money and Cardinal types reinforce the findings that the subword tokenization in BERT is a suboptimal method to encode numerical values [61]. Overall our findings suggest that BERT could be more resistant to training data extraction attacks than GPT-2. This is most likely due to its smaller size as argued in [6], but it is also possible that Auto-Encoder Transformers are generally less prone to these attacks compared to Auto-Regressive Transformers as result of their different pre-training objectives.

Differentially Private Fine-tuning. In all the experiments that used Differentially Private fine-tuning, the extraction rates of named entities were reduced by a large extent. Our samples have shown that the text quality in the *DP* setups was very low, both text coherence and text diversity decreased dramatically. Even though the performance on the downstream task was also considerably lower, we argue this trade-off between performance and privacy is still promising. Considering that the focus of our study was not on achieving state-of-the art performance for singe-label text classification, we only used the original version of the DPSGD algorithm [1]. More recent DP algorithms offer a more modest performance decrease on many NLP tasks, while achieving the same or better privacy budgets [68]. One can expect that for tasks, where the ability of a model to generate text is irrelevant, the use of DP can be a viable solution to increase the privacy of the model.

6.2 Generalization

We expect that similar observations would be made on other text-classification datasets, as we have intentionally covered a wide range of characteristics in our two datasets. While the Enron dataset we used is small in size and is very cluttered due to its source (real world emails), the Blog Authorship Corpus contains a large amount of samples that cover a broad range of domains with a higher text quality. Although, We only used single-label text classification as a downstream task for fine-tuning, results should be similar on different downstream tasks since the memorization takes place in the encoding layers, irrespectively of the task specific final layers of a model. Finally our conclusion about the memorization capabilities of the BERT base model is in line with the training data extraction study done on Clinical BERT [33], in which the authors were unable to reliably extract patient names from a BERT variant pre-trained on clinical data.

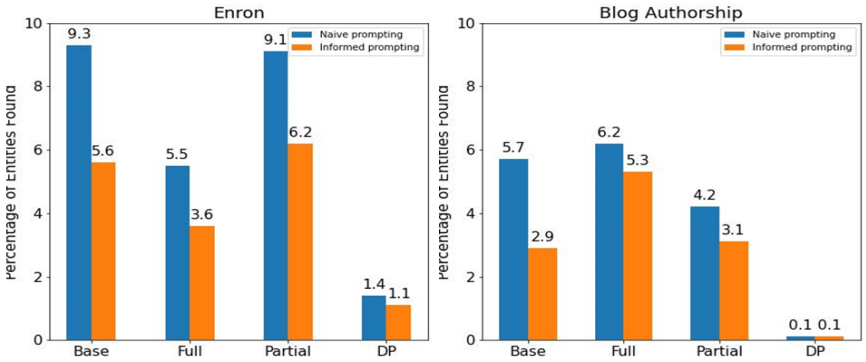

Figure 3: The percentages of all entities successfully extracted from the models, compared by prompting methods. In each column pair the left bar displays naive prompting and the right bar the informed prompting. The percentages are calculated based on the numbers shown in Table 2

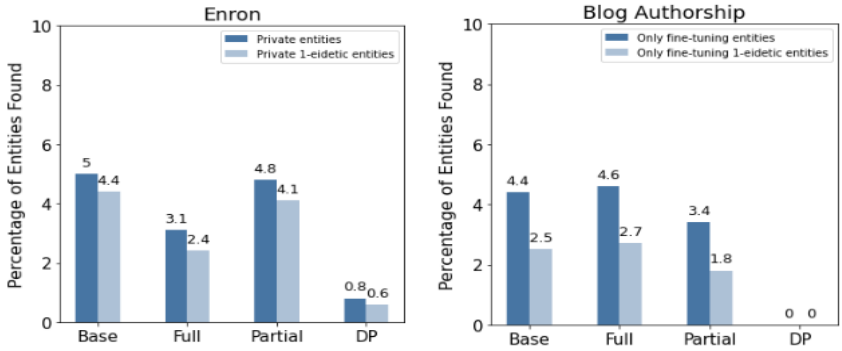

Figure 4: The percentages of private entities and private 1-eidetic entities successfully extracted from the models with the use of naive prompting. In each column pair the left bar displays the results for the *Private* entity set and the right bar shows the results for the *Private 1-eidetic* entity set. The percentages are calculated based on the numbers shown in Table 2

6.3 Threats to Validity

We acknowledge that the experimental datasets are limited to English. Although named entities are often unique to their respective language, we have no reason to believe that generating named entities would be significantly easier in different languages, and in some languages that have larger character sets (e.g., Chinese) or use long compound words (e.g., German) the probability of unintended memorization may be even more unlikely. Regarding the efficiency of our extraction of named entities, the results can be influenced by both the named entity recognition system and our text generation method. Although we used the state-of-the-art named entity recognition model from spaCy, it is highly possible that some entities have been missed and some have been identified falsely. The missed entities are unlikely to influence the results since we still had a great amount of entities of differing k-eidetic values. Controlling for the falsely identified entities was a more difficult problem, therefore we decided to remove all entities with a character length of less than 4. While this does not solve the issue completely, we assume that the remaining false positives did not hinder the study of unintended memorization. Using a left-to-right sequential text generation method might also bias our results, as BERT uses context from both directions to predict a token during pre-training. This, we argue is has more impact on text coherence rather than the ability to trigger a diverse output containing named entities, which was of higher importance to our study.

6.4 Practical Impact and Future Work

Our study offers insights into the privacy risks involved in employing a pre-trained BERT on private fine-tuning data. Although our results do not rule out the possibility to extract personal information from a fine-tuned BERT base model using more advanced methods, our findings suggest that doing so is at least not trivial. As for future work it would be interesting to re-run the experiments on BERT large and other commonly used BERT variants [30, 37, 53]. Even though our experiments revealed surprisingly minimal privacy leakage in BERT models fine-tuned by standard methods, it would be interesting to analyze the effects of Differential Privacy in other language models that are more prone to training data extraction attacks [6]. Another interesting area of future research would be to test the embedding layers of our BERT setups against membership inference attacks [38].

7 CONCLUSION

We performed an initial investigation into the capabilities of BERT to memorize named entities. We ran experiments, in which we tried to extract private named entities from fine-tuned BERT models using three different fine-tuning methods and two prompting strategies. Overall, our results show that only a low percentage of named entities can be extracted via our methods, and that an only pre-trained model is capable to generate the same amount of named entities. This suggests that the BERT base model might lack the capability to memorize named entities to the extent that they could be reliably extracted via training data extraction attacks. Additionally, we also employed a Differential Private fine-tuning method, which showed to be a promising privacy preserving method against training data extraction attacks. To facilitate further research, we make our experimental setups and baseline models available: https://github.com/drndr/thesis

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.

[2] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–35.

[3] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. 2017. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security* 13, 5 (2017), 1333–1345.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*. 267–284.

[6] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*. 2633–2650.

[7] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter* 19, 2 (2017), 25–35.

[8] Ali Davody, David Ifeoluwa Adelani, Thomas Kleinbauer, and Dietrich Klakow. 2020. Robust differentially private training of deep neural networks. *arXiv preprint arXiv:2006.10919* (2020).

[9] Emiliano De Cristofaro. 2020. An overview of privacy in machine learning. *arXiv preprint arXiv:2005.08679* (2020).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[11] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305* (2020).

[12] Christophe Dupuy, Radhika Arava, Rahul Gupta, and Anna Rumshisky. 2021. An efficient DP-SGD mechanism for large scale NLP models. *arXiv preprint arXiv:2107.14586* (2021).

[13] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our data, ourselves: Privacy via distributed noise generation. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 486–503.

[14] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407.

[15] Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science* 14, 2 (1990), 179–211.

[16] Wikimedia Foundation. [n.d.]. *Wikimedia Downloads*. https://dumps.wikimedia.org

[17] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1322–1333.

[18] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. 2014. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*. 17–32.

[19] Markus Freitag and Yaser Al-Onaizan. 2017. Beam Search Strategies for Neural Machine Translation. In *Proceedings of the First Workshop on Neural Machine Translation*. 56–60.

[20] Lukas Galke and Ansgar Scherp. 2022. Bag-of-words vs. graph vs. sequence in text classification: Questioning the necessity of text-graphs and the surprising strength of a wide MLP. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 4038–4051.

[21] Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), 721–741.

[22] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. *arXiv preprint arXiv:1904.09324* (2019).

[23] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International conference on machine learning*. PMLR, 201–210.

[24] Fadi Hassan, Josep Domingo-Ferrer, and Jordi Soria-Comas. 2018. Anonymization of unstructured data via named-entity recognition. In *International conference on modeling decisions for artificial intelligence*. Springer, 296–305.

[25] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751* (2019).

[26] Matthew Honnibal, Ines Montani, Sofie Van Landeghe, and Adriane Boyd. 2022. spaCy: Industrial-strength Natural Language Processing in Python". https://zenodo.org/record/121230

[27] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146* (2018).

[28] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[29] Bryan Klimt and Yiming Yang. 2004. Introducing the Enron corpus. In *CEAS*.

[30] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* (2019).

[31] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[32] Jaejun Lee, Raphael Tang, and Jimmy Lin. 2019. What would Elsa do? freezing layers during transformer fine-tuning. *arXiv preprint arXiv:1911.03090* (2019).

[33] Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron C Wallace. 2021. Does BERT Pretrained on Clinical Notes Reveal Sensitive Data? *arXiv preprint arXiv:2104.07762* (2021).

[34] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Sasko, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clement Delangue, Theo Matussiere, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, Francois Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A Community Library for Natural Language Processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 175–184.

[35] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. 2021. When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–36.

[36] Kin Sum Liu, Bo Li, and Jie Gao. 2018. Generative model: Membership attack, generalization and diversity. *CoRR, abs/1805.09898* (2018).

[37] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[38] Saeed Mahloujifar, Huseyin A Inan, Melissa Chase, Esha Ghosh, and Marcello Hasegawa. 2021. Membership Inference on Word Embedding and Beyond. *arXiv preprint arXiv:2106.11384* (2021).

[39] Huanru Henry Mao. 2020. A survey on self-supervised pre-training for sequential transfer learning in neural networks. *arXiv preprint arXiv:2007.00800* (2020).

[40] H Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629* (2016).

[41] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 691–706.

[42] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh. 2020. Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254* (2020).

[43] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes* 30, 1 (2007), 3–26.

[44] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 280–290.

[45] Seong Joon Oh, Bernt Schiele, and Mario Fritz. 2019. Towards reverse-engineering black-box neural networks. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 121–144.

[46] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).

[47] Mathias PM Parisot, Balazs Pejo, and Dayana Spagnuelo. 2021. Property Inference Attacks on Convolutional Neural Networks: Influence and Implications of Target Model's Complexity. *arXiv preprint arXiv:2104.13061* (2021).

[48] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018). https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

[49] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[50] Maria Rigaki and Sebastian Garcia. 2020. A survey of privacy attacks in machine learning. *arXiv preprint arXiv:2007.07646* (2020).

[51] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how bert works. *Transactions of the Association for Computational Linguistics* 8 (2020), 842–866.

[52] Benjamin IP Rubinstein, Peter L Bartlett, Ling Huang, and Nina Taft. 2009. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *arXiv preprint arXiv:0911.5708* (2009).

[53] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[54] J Schler, M Koppel, S Argamon, and JW Pennebaker. 2006. Effects of Age and Gender on Blogging in Proceedings of 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs. In *Proceedings of 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*.

[55] Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The cost of training NLP models: A concise overview. *arXiv preprint arXiv:2004.08900* (2020).

[56] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*. IEEE, 3–18.

[57] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.

[58] Om Thakkar, Swaroop Ramaswamy, Rajiv Mathews, and Françoise Beaufays. 2020. Understanding unintended memorization in federated learning. *arXiv preprint arXiv:2006.07490* (2020).

[59] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing Machine Learning Models via Prediction APIs. In *25th USENIX security symposium (USENIX Security 16)*. 601–618.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[61] Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP Models Know Numbers? Probing Numeracy in Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 5307–5315.

[62] Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: Bert as a markov random field language model. *arXiv preprint arXiv:1902.04094* (2019).

[63] Binghui Wang and Neil Zhenqiang Gong. 2018. Stealing hyperparameters in machine learning. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 36–52.

[64] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3454–3469.

[65] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*. 38–45.

[66] Yang Xu, Tinghuai Ma, Meili Tang, and Wei Tian. 2014. A survey of privacy preserving data publishing using generalization and suppression. *Applied Mathematics & Information Sciences* 8, 3 (2014), 1103.

[67] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, et al. 2021. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298* (2021).

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

[69] Santiago Zanella-Beguelin, Lukas Wutschitz, Shruti Tople, Victor Ruhle, Andrew Paverd, Olga Ohrimenko, Boris Kopf, and Marc Brockschmidt. 2020. Analyzing information leakage of updates to natural language models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 363–375.

[70] Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337* (2022).

[71] Jiajun Zhang, Chengqing Zong, et al. 2015. Deep Neural Networks in Machine Translation: An Overview. *IEEE Intell. Syst.* 30, 5 (2015), 16–25.

[72] Tianwei Zhang, Zecheng He, and Ruby B Lee. 2018. Privacy-preserving machine learning through data obfuscation. *arXiv preprint arXiv:1807.01860* (2018).

[73] Jingwen Zhao, Yunfang Chen, and Wei Zhang. 2019. Differential privacy preservation in deep learning: Challenges, opportunities and solutions. *IEEE Access* 7 (2019), 48901–48911.

[74] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *The IEEE International Conference on Computer Vision (ICCV)*.

Supplementary Materials

A EXTENDED DATASET PREPROCESSING

The following sections contains additional details about the preprocessing. In particular, we give further information about the preparation of the Enron dataset, and provide an overview of the label distributions.

A.1 Preprocessing the Enron dataset

As mentioned in Section 4.1, the Enron dataset does not come with categorical labels. The raw dataset contains the text of the email messages, alongside additional headers with meta-information. One of these headers called "X-folder" contains the location of the email in folder structure of each user. We used the last folder name in the location paths to create labels. While these folders are unique to each user, some general overlap between the naming conventions and folder contents made it possible to select 7 folders as labels that can be used for single-label text classification. These labels include: "logistics", "personal", "management", "deal discrepancies", "resumes", "online trading", and "corporate".

During preprocessing the text of the emails, we found that the message bodies contained a great amount of message chains, where some meta-information from the email headers (such as lists of email addresses, network information, message ID) were also present between the texts. This turned out to have a negative effect on output of our models when generating text. Namely, the generated samples lost all coherence and contained a lot of random concatenation of subword tokens. Therefore, we decided to discard all reply and forward chains from the messages by removing the parts following phrases that indicate a replied or forwarded message. This additional preprocessing, alongside the removal of HTML links substantially improved the the diversity and quality of the generated text samples.

A.2 Label Distributions

Figure 5 and 6 display the label distribution of both datasets. In the Enron the most frequent label is "personal" with 2,062 occurences, while on the Blog Authorship dataset the label "Student" has the highest value count with 153,903. During the train-test split, the distribution of the dataset was retained in both splits.

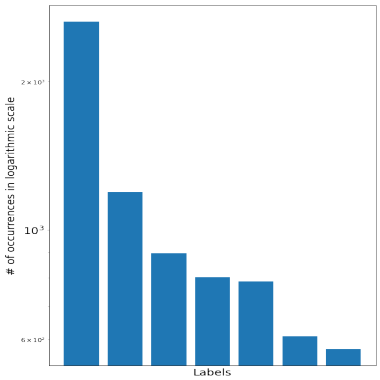

Figure 5: Label distribution of the Enron dataset

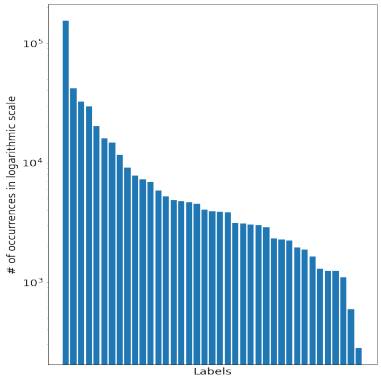

Figure 6: Label distribution of the Blogauthorship dataset

B EXTENDED EXPERIMENT RESULTS

This Section contains the extended experimental results. We provide an extended review of the NER results with detailed description of the entity types, a discussion about selecting specific entity types for our study, and the additional named entity extraction results not present in the main paper.

B.1 Named Entity Recognition

As mentioned in Section 4.2 we used the spaCy library [**?**] for the process of NER. The named entity recognizer of spaCy distinguishes between the following 18 entity types:

- **PERSON** - people names, including fictional
- **NORP** - nationalities or religious or political groups
- **FAC** - facilities - building, airports, bridges, etc.
- **ORG** - organizations,companies, agencies, institutions
- **GPE** - geopolitical entities - countries, cities, states
- **LOC** - non-GPE locations
- **PRODUCT** - objects, vehicles, foods
- **EVENT** - named hurricanes, wars, sport, events etc
- **WORK_OF_ART** - titles of books, songs, etc
- **LAW** - named documents made into laws
- **LANGUAGE** - any named language
- **DATE** - absolute or relative dates, periods
- **TIME** - times smaller than a day
- **PERCENT** - percentages
- **MONEY** - monetary values, including unit
- **QUANTITY** - Measurements, as of weight or distance
- **ORDINAL** - "first", "second", etc
- **CARDINAL** - numerical values that do not fall under other type

The initial results of the NER can be seen in Table 6. Compared to the values in Table 2, these counts also include the repeated occurrences of a named entity.

Table 6: Original number of named entities in the datasets, sorted by type

Named Entity Type	Enron	Blog Authorship
PERSON	33,993	767,144
NORP	1,373	132,080
FAC	674	24,405
ORG	30,365	521,159
GPE	8,771	298,607
LOC	827	37,286
PRODUCT	992	27,432
EVENT	333	14,131
WORK_OF_ART	1,354	43,102
LAW	151	5,798
LANGUAGE	69	11,161
DATE	17,319	626,969
TIME	7,137	211,993
PERCENT	579	17,332
MONEY	2,952	39,768
QUANTITY	655	21,512
ORDINAL	1,210	104,559
CARDINAL	17,725	444,764

In studying the extent of named entity memorization, we focused on entity types that have a higher probability to contain personal or privacy sensitive information. *PERSON* can include first and last names, which together can be considered as personal information. *ORG*, *GPE*, *LOC* and *FAC* include information that can be pieced together to identity a likely data subject, therefore they also fit into the personal information category. We included *Money*, as in some industries (i.e., banking) records of specific amounts can be regarded as privacy sensitive information. Finally, we also included *CARDINAL* as this type can refer to card numbers, phone numbers and different ID number, which can be both personal and privacy sensitive.

B.2 Named Entity Memorization

The following Section includes the results of the experimental setups not shown in the main text. The percentage values are based on the number shown in Table 5.

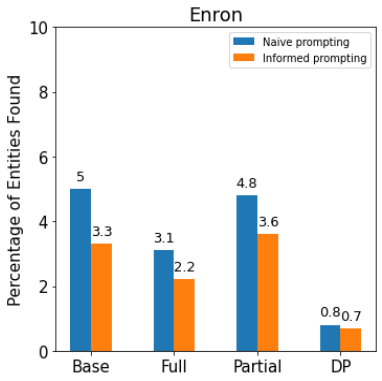

Figure 7: The percentages of private entities extracted from the models, compared by prompting methods.In each column pair the left bar displays naive prompting and the right bar the informed prompting. The percentages are calculated based on the numbers shown in Table 2

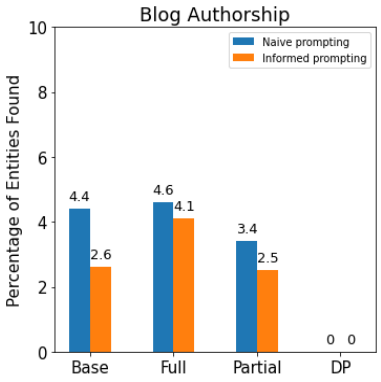

Figure 8: The percentages of private entities extracted from the models, compared by prompting methods.In each column pair the left bar displays naive prompting and the right bar the informed prompting. The percentages are calculated based on the numbers shown in Table 2

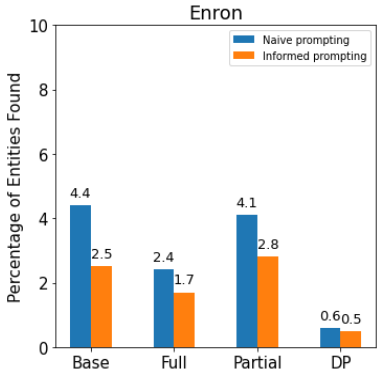

Figure 9: The percentages of private 1-eidetic entities extracted from the models, compared by prompting methods.In each column pair the left bar displays naive prompting and the right bar the informed prompting. The percentages are calculated based on the numbers shown in Table 2

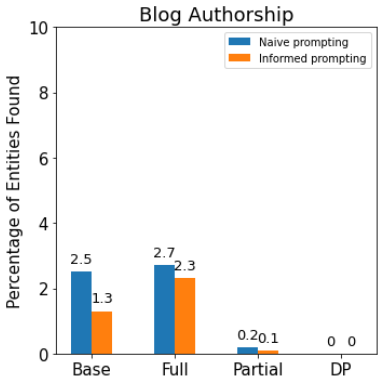

Figure 10: The percentages of private entities extracted from the models, compared by prompting methods.In each column pair the left bar displays naive prompting and the right bar the informed prompting. The percentages are calculated based on the numbers shown in Table 2