universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Sensitive Information Leakage in Neural Taggers“

verfasst von / submitted by

Lisa Maria Nußbaumer, BA MA

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Arts (MA)

Wien, 2023 / Vienna, 2023

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 647
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Digital Humanities
Betreut von / Supervisor:	Univ.-Prof. Dr. Benjamin Roth

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. Benjamin Roth and his NLP working group, whose expertise and guidance were crucial in writing this thesis. Special thanks to Yuxi Xia for her valuable contributions and dedication to support my research. I would like to extend my heartfelt thanks to the RG head of my team at Siemens, Herwig Schreiner, for providing me with the freedom to pursue this research. In this context, I would also like to acknowledge the ongoing encouragement and support of my colleagues at work throughout the process of writing this thesis. Finally, I am deeply grateful to my partner Christoph, family, and friends, who kept me motivated and helped me to balance my personal, professional and academic responsibilities with ease by keeping me grounded in what truly matters in life.

Abstract

The enhanced performance of recent Large Language Models (LLMs) in different types of natural language tasks has revolutionized the field of artificial intelligence and accelerated the adoption of LLM-based applications in various domains. However, amidst the hype surrounding LLMs, security-related aspects, such as potential risks for data leaks, should not be neglected, as LLMs are typically trained on vast amounts of data, which may contain sensitive information. In particular, the potential for sensitive data leakage presents a growing concern for society as a whole considering the increased pervasiveness of LLMs. Therefore, the aim of this thesis is to develop a risk analysis framework for the leakage of sensitive information from the training data of LLMs. This framework involves several black-box attack strategies to estimate the information leakage of a Named Entity Recognition model and is based on the assumption that data instances that were used for the training of the model receive higher prediction scores than those that were not included in the training dataset. The findings in this thesis indicate that the output score of a neural tagger can serve as a reliable indicator to detect and estimate sensitive information leakage from a LLM. Furthermore, additional research questions have been raised addressing the influence of model scaling and prompt engineering on data leakages from LLMs, but require further investigation which extends beyond the scope of this thesis.

Kurzfassung

Die gesteigerte Leistungsfähigkeit von aktuellen, großen Sprachmodellen (Large Language Models, LLMs) für unterschiedliche Aufgaben in der natürlichen Sprachverarbeitung haben den Bereich der Künstlichen Intelligenz revolutioniert und die Einführung von LLM-basierten Anwendungen in vielen Domänen beschleunigt. Inmitten dieses Hypes rund um LLMs darf aber nicht auf sicherheitsrelevante Aspekte, wie potenzielle Risiken für Datenlecks, vergessen werden, da LLMs üblicherweise mit sehr großen Datenmengen, die unter anderem sensible Daten beinhalten könnten, trainiert werden. Betrachtet man die schnelle Verbreitung von LLMs, bereitet besonders das Risiko, dass sensible oder persönliche Daten offengelegt werden könnten, erhebliche Bedenken. Daher ist das Ziel dieser Arbeit, ein Risikoanalysekonzept für Datenlecks aus den Trainingsdaten von LLMs zu entwickeln. Dieses Konzept umfasst mehrere Black-Box Angriffsstrategien, um das Datenleakpotenzial eines Eigennamenerkennungs- (Named Entity Recognition, NER) Modelles abzuschätzen und basiert auf der Annahme, dass Dateninstanzen, die für das Training des Modells verwendet wurden, höhere numerische Vorhersagewerte erhalten als solche, die nicht im Trainingsdatensatz inkludiert waren. Die Erkenntnisse dieser Arbeit zeigen, dass Ausgabewerte eines neuronalen Taggers als zuverlässiger Indikator dienen können, um Datenlecks in einem LLM zu erkennen und zu bewerten. Darüberhinaus wurden weitere Forschungsfragen aufgeworfen, die den Einfluss von Modellskalierung und Prompt-Engineering auf Datenlecks in LLMs thematisieren, aber weitere Untersuchungen erfordern, die über den Rahmen dieser Arbeit hinausgehen würden.

Contents

List of Tables

List of Figures

Acronyms

AI Artificial Intelligence. 7, 22

AP Average Precision. x, 35, 47, 51

BERT Bidirectional Encoder Representations from Transformers. 1, 10, 12, 39, 55

CCPA California Consumer Privacy Act. 9

DPO Data Protection Officer. 9

ELMo Embeddings from Language Models. 11

FN False Negative. 15

FP False Positive. 15, 39, 40

GDPR General Data Protection Regulation. 7, 8

GPT Generative Pre-trained Transformer. 1, 11, 14, 20, 22

HIPAA Health Insurance Portability and Accountability Act. 8

LGPD Lei Geral de Proteção de Dados. 9

LLM Large Language Model. 1, 10, 12

LLMs Large Language Models. iii, 1, 7, 12, 16, 17, 39, 53–55

LM Language Model. 10, 12

LSTM Long Short Term Memory. 10

MAP Mean Average Precision. x, 35, 47, 51

MLM Masked-Language-Modeling. 10, 13, 22

NE Named Entity. 14

NER Named Entity Recognition. iii, ix, 10, 11, 14, 15, 21, 22, 55

NLP Natural Language Processing. 1, 10–12, 17

NSP Next Sentence Prediction. 10, 13, 22

PER Person. 3, 9, 25, 39, 55

PHI Personal health information. 7

PII Personally Identifiable Information. 7–9, 14, 17, 29, 54, 55

POS Part of Speech. 24

TP True Positive. 15, 39

Nomenclature

μ_{neg_i}	averaged PER-tag prediction score for multi-token negative sample
μ_{pos_i}	averaged PER-tag prediction score for multi-token positive sample
ϕ	prediction score (numerical)
A_{dev}	development data set for attack
A_{test}	test data set for attack
C	corpus of input prompts c_i
c_i	input prompt for attack
$D\prime$	data set which includes a subset of D_{train}
D_{train}	training data set of a model M
Kn	sum of wins for negative sample in majority voting
Kp	sum of wins for positive sample in majority voting
n_{neg}	number of negative samples
n_{pos}	number of positive samples
n_{word}	number of words
neg_i	negative sample
pos_i	positive sample
s_i	sample name from set A_{dev} or A_{test}
w_i	target data points (words or tokens)

1. Introduction

The past decade has seen a rapid development of language models for a wide range of different Natural Language Processing (NLP) tasks. Especially, in the field of deep learning, there has been a focus on increasing the size and complexity of neural network architectures in order to improve the performance of models, which can identify, predict and generate natural language data. Just in the past five years, several advanced Large Language Models (LLMs) such as BERT [Devlin et al., 2018] and the GPT-series [Radford et al., 2019, Brown et al., 2020, OpenAI, 2023] have been developed and demonstrated a strong performance across a broad range of domains and applications. As a result, these models are being used in a variety of language modeling solutions beyond just academic research, for example, in the form of neural machine translation [Dabre et al., 2020] or chatbots [Adamopoulou and Moussiades, 2020], which have received particular attention from the public since the release of ChatGPT and other dialogue-based AI systems. It is thus important to study the effects on security and privacy in machine learning as language models become more widely integrated into daily life.

A significant potential vulnerability in terms of privacy for neural language models is their training data. Training data is the first and essential component in the process of developing a language model. LLMs (mostly) use text-based datasets to learn probabilities of word occurrences or specific language properties and transfer the learned knowledge onto unseen texts [Hiemstra, 2009]. By its nature, a LLM commonly requires large amounts of training data. To acquire these large quantities of training data, examples are often collected through the process of crawling publicly available texts on the internet or using private datasets. This data can, however, contain sensitive information of various kinds, such as names, unique identifiers or confidential data. Ideally, a model should not memorize or disclose any sensitive information that was part of the training corpus, but in practice, research has shown that models can be attacked to output training data instances ([Carlini et al., 2018] and [Thomas et al., 2020], amongst others). This undesired sharing of data during the deployment of a model is known as information leakage and might pose security risks for or infringe privacy rights of individuals and organizations.

1.1. Motivation

The release of BERT marked the beginning of a new era in language modeling research, characterized by the use of larger models and larger datasets to achieve new benchmarks in performance and opening up new areas of application. This shift towards bigger models has had a significant impact on the field and has recently raised important considerations about data safety and privacy due to the potential for training data leakage [Brown et al., 2022, Balle et al., 2022].

As the size of datasets used to train LLMs continues to grow, there is a greater risk that sensitive information may be included in the training data. This is often due to the high cost of properly curating such large internet-search-based datasets, which can make it difficult to thoroughly review and remove sensitive information [Bender et al., 2021]. Therefore, investigating possible factors for training data leakage as well as its extent constitutes a major area of interest in the field of NLP research.

Privacy attacks on LLMs can take many forms, yet they all rely on the requirement that the targeted language model and/or its parameters are available to or accessible by attackers. While it is important to share research efforts and breakthroughs, data privacy should not be neglected in this context. The possibility of privacy breaches when sharing (pre-trained) models or their training parameters has already received attention from research communities, particularly those involved in handling privacy sensitive data such as, e.g., biomedical NLP [Xafis et al., 2019, Nakamura et al., 2021]. An attack on a model trained on patient data could have serious consequences for individuals and society. For example, [Lehman et al., 2021] call attention to a worst-case scenario in which leaked patient information may lead to the rejection of mortgage or credit card applications for individuals. This means that revealing the mere presence of certain data points in a dataset along with the intended use of the model can have negative consequences, including the compromise of personal privacy and the violation of ethical norms. Therefore, it is important to identify and mitigate the risk[1] of sensitive information leakage in language models in order to ensure their responsible and ethical use.

[1] In a different context, the leakage of sensitive information used for training a language model might not be considered a risk, but rather as a chance to expose the unauthorized use of sensitive information during the training process of models as [Rigaki and Garcia, 2020] argue. However, this alternative approach does not fall under the objectives of this thesis.

1.2. Problem Definition

The problem addressed in this thesis is the leakage of sensitive information, in specific Person (PER) names, which were used to train a neural language model. Despite the fact that awareness regarding data (set) privacy has increased, there exists no commonly accepted test or metric to assess the training data leakage of LLMs [Tamkin et al., 2021]. One possible reason for the lack of an one-fits-all solution for evaluating data leakage in LLMs could be the diversity of training processes and deployment strategies. The development of a comprehensive and unified solution for data leakage assessment lies beyond the scope of this thesis, but instead this research will specifically focus on developing an analytical framework to examine training data leakage of a sequence labeling model, because sequence labeling is an integral part of many NLP applications, enabling systems to extract and identify relevant pieces of information present in linguistic data [Jurafsky and Martin, 2023].

A broader term for sequence labeling models are neural taggers, which additionally specify that the model is based on an artificial neural network. While sequence labeling refers more explicitly to the process of assigning labels to individual elements (words, phrases etc.) based on their grammatical function or role in a sequence, tagging applies to any process of annotation and thus encompasses sequence labeling [Rei et al., 2016]. In this thesis, the terms *sequence labeling* and *tagging* will be used interchangeably, as they are both closely related concepts in NLP.

The main aim of this research is to deepen the understanding of data leakage in neural tagger language models and develop new methods for evaluating it. To achieve this, the study uses probing attacks, which involve PER names as target data points w_i to determine whether w_i is a member of a training dataset D_{train} of a model M. Following traditions in information security research literature, three characters involved in a possible data leakage scenario are introduced:

(a) **Alex** (model& data owner): knowledgable about the training dataset D_{train} and model M; has access to both.

(b) **Blake** (adversary): Blake uses the prediction score $\phi(w_i)$ for the label tag PER outputted by the Model M to determine if $w_i \in D_{train}$. Blake has only query access to the model;

(c) **Charlie** (baseline adversary): Charlie follows the strategy of stratified random guessing. In the discussed case, this corresponds to flipping a fair coin, i.e. 50:50 chance

of being a member or non-member of the training dataset D_{train}. This serves as a baseline to compare the attack results of (b).

In the outlined scenario, *Alex*, *Blake* and *Charlie* work together on the question of how probing attacks can be used to evaluate the leakage of PER names from neural taggers in an analytical risk framework. Their aim is to check the model's robustness against privacy attacks. Therefore, they have assigned roles as (a) model and data owner, who knows the PER names in the the training dataset D_{train} and adversaries, who (b) exploit the model output for insight or (c) use a rather basic approach of stratified random guessing. An adversary is an attacker or malicious threat actor who seeks to cause harm or exploit vulnerabilities in a system or network [Anderson, 2008]. A more detailed description of the used model, its training dataset and the attack strategies is provided in Chapter 3.

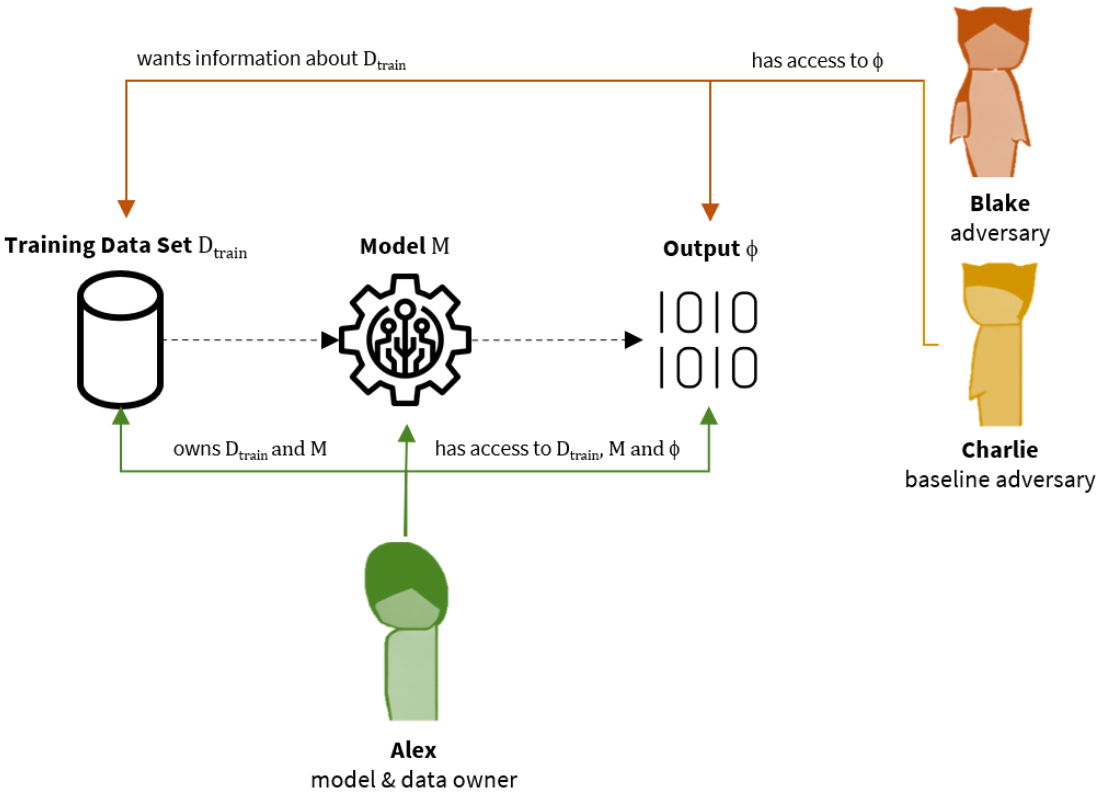

Figure 1.1.: Schematic view of attack scenario. The character symbols were generated by DALL-E Demo licensed under CreativeML Open RAIL-M.

1.3. Contribution

This thesis extends previous research by providing a thorough investigation into the influence of prediction scores generated by neural tagger models regarding the leakage of sensitive information from the model's training data. In this context, an analytical framework to assess the risk of training data leakage is developed by testing various methods and evaluation metrics. This work seeks to contribute to the growing research field of identifying information leakage in LLMs by exploring a novel measure to quantify the scope of privacy attacks on neural taggers. The measure of expected rank, $E(rank|w_{train})$, defines a conditional expectation value for the rank of a randomly selected training data instance in a ranked list of possible training data candidates. In a simulated privacy attack on the target model, the proposed analytical framework involving the expected rank along with traditional metrics, such as accuracy and average precision, can be used as leakage indication for language models based on their prediction output.

1.4. Outline

This thesis is organized into five main chapters. The following chapter begins by briefly examining the concept of sensitive information and explains why protecting this kind of data is crucial. Chapter 2 provides a detailed overview of the existing literature on information leakage in LLMs, including key studies and findings in this area. Chapter 3 describes the research design and methods used in the study, including the resources used, such as the neural tagger model and the datasets used for the attack scenario, as well as the metrics used to measure leakage risk. The fourth chapter presents and analyzes the results of the attack experiment and discusses the performance of the strategies proposed in the analytical risk framework. The final two chapters, Chapter 5 and 6, of the thesis examine the limitations of the study, identify opportunities for further research, summarize the key findings and provide a general evaluation of the research problem.

2. Background and Related Research

This section provides a definition of sensitive information and discusses the protection status of this kind of data from a legal perspective on the example of the General Data Protection Regulation (GDPR). The second part of this chapter gives a brief overview on the recent history of (Large) Language Models and explains the basic training procedure of LLMs and the data used in this context. The chapter concludes with an outline of different types of privacy attacks.

2.1. What is Sensitive Information?

As the use of Artificial Intelligence (AI) systems trained on real-world data becomes more prevalent, the potential for personal information to be exposed also increases. Due to advancements in technology that allow for storage, processing and analysis of vast amounts of data and the facilitated access to an ever-increasing supply of data online, more and more data, including sensitive information, is being collected and used for training of machine learning models. Sensitive information refers to data that requires a high level of protection due to its nature and the potential negative impact of unauthorized disclosure or access [Quinn and Malgieri, 2021]. Sensitive data can be in the form of text, images, or other types of (digital) files. The definition of sensitive data varies depending on the context, but it typically includes personal information such as names, medical records, proprietary information along with other confidential data [Quinn and Malgieri, 2021]. In academic literature, sensitive data is often described as information that falls into one or more of the following categories:

- Personally Identifiable Information (PII), which can include names, addresses, social security numbers, and other data that can be used to identify an individual [Lison et al., 2021].
- Personal health information (PHI)[1], which refers to information related to an indi-

[1] There exists an alternative compound term for PHI: *Protected Health Information*. It is mainly used in the

vidual's physical or mental health, including medical records, treatment plans, and other sensitive medical information [Winter and Davidson, 2019].

— Financial information, such as credit card numbers, bank account information, and other financial records [Scholz, 2019].

— Intellectual property, such as trade secrets, patents, and other proprietary information that could provide a competitive advantage if it were to be revealed [Bhilare et al., 2009].

There have been a variety of legal efforts aimed at protecting data privacy. For example, the European Union's General Data Protection Regulation (GDPR) sets strict guidelines for the collection, use, and storage of personal data, including the requirement to obtain explicit consent for the use of personal data in machine learning [Murakonda and Shokri, 2020]. However, [Brown et al., 2022] note that, in the case of publicly available data used for machine learning, for example social media data, individual data subjects cannot properly consent to the use of their data. Furthermore, obtaining consent for the use of this kind of data would involve an immense administrative effort by the data processing party.

The GDPR gives a very broad definition of Personally Identifiable Information in Art. 4(1) "Personal data are any information which are related to an identified or identifiable natural person." [European Commission of Justice and Consumers, 2016]. This comprehensive interpretation of PIIs has a significant impact on machine learning, as personal or sensitive data may be included in the input used to train and test machine learning models. The use of sensitive data in the training of these models has been a major area of concern because a breach of the data used to train models could possibly lead to the exposure of sensitive information [Nasr et al., 2019]. For example, the occurrence of PIIs in training datasets can result in the re-identification of individuals, which is particularly concerning when the data contains sensitive information such as medical records or financial data. The leakage of this type of data may not necessarily be caused by an attack on the model, but rather by the model's failure to understand the concept of data sensitivity [Brown et al., 2022]. If not annotated in some form, it is not evident for a model, which instances in the training dataset classify as sensitive information.

To effectively manage large amounts of data while simultaneously ensuring the security of the data used to train, test and deploy machine learning models, the GDPR introduced

context of Health Insurance Portability and Accountability Act (HIPAA) and is synonymous with Personal Health Information.

the role of a Data Protection Officer (DPO) responsible for the compliance of the regulation (Art. 38 and Art. 39). The DPO plays a crucial role in the GDPR framework as the main point of contact for data subjects, the supervisory authority, and the public. They are responsible for administering data protection risks and identifying and reporting data breaches. It is important to note that the GDPR's legal framework is not the only effort to address data protection in the context of data processing. Other similar (inter)national legal efforts have also been implemented, such as the California Consumer Privacy Act (CCPA) in the United States and the Lei Geral de Proteção de Dados (LGPD) in Brazil. Together, these legal frameworks demonstrate that data protection has been acknowledged as an important aspect of machine learning. This is because they safeguard sensitive information from being compromised during model training or deployment.

In addition to these legal actions, there are also ethical guidelines and best practices that have been developed for the use of machine learning ([Boddington, 2017, Brundage et al., 2018, Jobin et al., 2019], amongst others). These guidelines can help organizations and institutions to ensure that they are using machine learning in a way that is fair, transparent, and respects individuals' privacy rights. In sum, these efforts stress the significance of not only protecting data but also of identifying possible data leakages in machine learning at an early stage in order to mitigate risks and ensure ethical and safe use of data and technology.

This thesis narrows its focus on the category of Personally Identifiable Information, in specific, its subcategory of *direct identifiers*, which includes PER names. Following a definition provided by [Lison et al., 2021], direct identifiers can be specified as "[a] (set of) variable(s) unique for an individual (a name, address, phone number or bank account) that may be used to directly identify the subject." Additionally, the research is limited to PER names consisting of more than one word. For instance, the two-word name *Robin Hood*, would be considered as a PER name, whereas the words *Robin* or *Hood* on their own are not regarded as a PER name in this analysis. The reason behind this restriction lies in the fact that single token name parts cannot be counted as unique identifiers of persons and therefore do not fall within a narrow interpretation of sensitive information. While there is a minimum limit for the number of words n_{word} in a PER name $n_{word} \geq 2$, there is no set upper limit.

2.2. Theory

This section is comprised of four parts, each addressing a different aspect of language modeling. The first part provides a general introduction to the broader topic of (Large) Language Models, with specific emphasis on the LLM BERT, which serves as the source model for the targeted NER model in the simulated attack scenario. The next section deals with the training process of LLMs and highlights The next section deals with the training process of LLMs and highlights several NLP tasks in this context, including, inter alia, Named Entity Recognition. This is followed by a section focused on the training data of LLMs. The final part explores potential privacy risks associated with LLMs and presents an overview of different privacy attacks.

2.2.1. (L)LMs

At its early stages, classical text analysis in machine learning primarily focused on word-based or phrase-based statistical modeling of natural language using n-grams [Brown et al., 1993] and Bag-of-Words models [McCallum et al., 1998]. Despite the limitations in computing power at the time, n-grams and Bag-of-Words models allowed for efficient processing and analysis of large amounts of data, motivating the development of Language Models (LMs) based on corpora [Clark et al., 2013]. However, these methods have constraints in capturing the semantics of words, leading to a shift towards deep learning frameworks, such as LSTM-based [Hochreiter and Schmidhuber, 1997] and transformer-based [Vaswani et al., 2017] LMs that aim at mapping the relationship between form and meaning of words. As a result, deep learning frameworks have emerged as the dominant approach for solving NLP tasks in LMs, achieving performance levels comparable to human capabilities [Min et al., 2021].

[Devlin et al., 2018]'s proposal of a neural network based on a transformer architecture titled BERT (Bidirectional Encoder Representations from Transformers) marked a significant advancement in the field of language modeling. BERT's unique bidirectional encoder representation, which allows for the parallel processing of both past and future context, resulted in a cutting-edge performance on a wide range of natural language processing tasks and greatly reduced training time, while outperforming other models [Min et al., 2021]. BERT is trained using a dual-phase method that incorporates transfer learning [Yang et al., 2020]: in phase one, a general model is pre-trained on a large, unlabeled dataset or corpus. This pre-training process uses techniques such as Masked-Language-Modeling (MLM) and and Next Sentence Prediction (NSP) to learn rich representations of word mean-

ing [Jurafsky and Martin, 2023]. In phase two, this pre-trained model is further used as a baseline model to learn specific NLP tasks with smaller datasets. By inserting the step of pre-training models, it is possible to obtain general language representations[2] that can be applied to a wide range of language modeling and understanding tasks, as for example NER, eliminating the need to start training a model from scratch [Qiu et al., 2020]. An addtional advantage of pre-training models according to [Erhan et al., 2010] is that it can also be seen as a regularization mechanism to some extent helping to prevent overfitting on small datasets. The upcoming section will elaborate further on the dual-stage training procedure of LLMs.

BERT's fundamental novelty is its advanced method of handling input bidirectionally. Unlike former mechanisms[3], such as one-directional LMs like OpenAI's GPT, which read text input sequentially from one direction, BERT processes the entire input sequence at once, allowing each token to pay attention to other tokens from both left to right and right to left. In the context of NLP, the term token refers to a sequence of characters that represents a single unit of meaning or grammatical structure, such as a character, word or phrase [Jurafsky and Martin, 2009]. The bidirectional processing is achieved through the use of a network of stacked encoders which apply multiple-head-self-attention[4], enabling BERT to learn different semantic meanings at each layer of the network [Devlin et al., 2018]. This breakthrough in training techniques and model architecture has generated a significant increase in interest and a plethora of investigations in this particular field of research.

Originally, BERT was published in two sizes: BERT_{Base} and BERT_{Large}. The size of the model is determined by three main hyper-parameters, which define (1) the number of encoder units (i.e. layers of the model), (2) the hidden size of each embedding vector[5]

[2] Following the definition of [Qiu et al., 2020], a good language representation should capture the implicit linguistic rules and common sense knowledge that are inherent in text data, including, but not limited to, lexical meanings, syntactic structures, semantic roles, and pragmatic implications.

[3] For the sake of completeness it has to be noted here that the model ELMo [Peters et al., 2018] is also described as bidirectional by its authors, but in a strict sense only qualifies as weakly bidirectional, because ELMo uses two models, one trained left-to-right and another trained right-to-left, which are then concatenated.

[4] Self-attention is a mechanism introduced by [Vaswani et al., 2017] used in deep learning to assign weights to different components of a model's input sequence based on their relevance to a particular context or language task [Raschka, 2023]. In multiple-head-self-attention the input sequence is transformed into several sub-representations, referred to as "heads", allowing neural networks to attend to different parts of the input in parallel [Vaswani et al., 2017].

[5] Embedding vectors serve as a means to represent the meaning of words and sentences and are derived from the semantic information in the training dataset [Brown et al., 2022]. As output of the pre-training, they can be further used as input in downstream NLP tasks.

and (3) the number of attention heads in each self-attention layer [Devlin et al., 2018]. An overview of the dimensions of BERT$_{Base}$ and BERT$_{Large}$ is given in Table 2.1.

Model Size	BERT$_{Base}$	BERT$_{Large}$
Layers	12	24
Hidden Size	768	1024
Self-attention heads	12	16
Total Parameters	110 Mio.	340 Mio.

Table 2.1.: BERT$_{Base}$ and BERT$_{Large}$ compared on the number of their hyperparameters.

At the time of its release, BERT was among the largest models trained, but is now considered small compared to newer models such as GPT-3 [Brown et al., 2020], Megatron [Smith et al., 2022] and PaLM [Chowdhery et al., 2022]. These models are significantly larger in terms of parameter count, with billions of parameters. The trend towards models with increasingly high numbers of parameters as well as the use of vast amounts of data during training has also reflected in the terminology used for these models as they have been termed as *Large* Language Models (LLM) [Tamkin et al., 2021]. In other words, the attribute "large" was added to the expression to distinguish more recent LMs from smaller, less complex models that have traditionally been used in NLP. Throughout this thesis the focus will be exclusively on the role of more recent language models, referred to as LLMs and their risk of leaking training data. Older language models, or LMs, will not be considered in this investigation.

2.2.2. Model Training

This section will give a comprehensive overview of the two key training phases of LLMs: pre-training and fine-tuning, with a specific focus on the BERT model. It will examine pre-training objectives, fine-tuning methods and the related types of NLP tasks.

In general, training is the process of adjusting the parameters of a model to optimize its performance on a given set of data, also known as training data [Bottou et al., 2016]. In neural networks, the parameters that need to be adjusted are the weights and biases of the different layers[6] of the network [Schmidhuber, 2015]. The training process typically involves initializing these parameters with random values and then modifying them through an optimization algorithm to minimize a loss function, which measures the difference

[6] In this context, a layer is a collection of neural units, which take real-valued data input, perform a specific computational operation and produce an output that is then passed forward in the network [Jurafsky and Martin, 2023].

between the model's predictions and the true values of the output [Dargan et al., 2020]. The final values of the weights after training represent the learned information of the network, which can be used to make predictions on new input data [Hintersdorf et al., 2021].

Pre-Training

The idea behind pre-training is that during this initial model training phase, the language model can learn general contextual representations of word meanings, which can be leveraged further to facilitate the learning of downstream language understanding tasks in the subsequent process of model fine-tuning [Jurafsky and Martin, 2023]. The pre-training is usually performed with a large, unlabeled dataset using self-supervised learning[7], while the fine-tuning is realized with a smaller, labeled dataset to adapt the model to a particular NLP task. This pretrain-finetune paradigm is an example of transfer learning, in which knowledge learned from one domain or task is applied (or transferred) to a new task [Jurafsky and Martin, 2023].

The pre-training of BERT involves two different language learning objectives: Masked-Language-Modeling (MLM) and Next Sentence Prediction (NSP). The main purpose of MLM is to train for bidirectionality [Devlin et al., 2018]. This is done by reserving 15%[8] of each training input sequence for masking before feeding the input into BERT. Of these randomly selected 15%, 80% are exchanged with a special [MASK] token, 10% are replaced with a randomly chosen different word the remaining 10% are left unchanged, i.e. with the original token. The model then attempts to predict the original value of the masked word without any information about the category it belongs to (80-10-10). The predictions are made based on the context provided by the other non-masked words in the sequence. In this way, the model takes into account both the context , i.e. tokens, on the right of the mask and the context, i.e. tokens, on the left of the mask. In contrast to MLM, which concentrates on word-level predictions, Next Sentence Prediction shifts the focus to sentence-level predictions. NSP can be understood as a binary classification task with the goal to predict whether two sentences are consecutive or not. The NSP component is intended to assist BERT in learning relationships between sentences [Devlin

[7] Self-supervised learning is a model training approach used when the training dataset lacks labels. In NLP this implies learning meaningful representations of the input text automatically without explicitly labeled data input [Bengio et al., 2013].

[8] [Devlin et al., 2018] argue that a masking rate of 15% strikes a balance between the challenge of training an excessively expensive model (with lower masking percentages resulting in fewer predictions) and the lack of context for training (with higher masking percentages resulting in poor performance).

et al., 2018]. Taken together, these pre-training objectives, MLM and NSP, enable BERT to acquire bidirectional representations of the input with a focus on the linguistic context. GPT-variants, on the other hand, adopt a different approach in pre-training. As they are generative LLMs they concentrate on predicting the next word in a sequence based on the previous words, which is a form of uni-directional training. In that way, GPT is able to produce coherent and fluent text [Brown et al., 2020].

Fine-Tuning

Following the pre-training phase, LLMs can be further optimized for new specific language tasks using either a fine-tuning or a feature-based approach. The fine-tuning strategy entails adjusting the parameters of the pre-trained model to maximize performance on a new task or dataset, whereas the feature-based approach involves using the pre-trained model as a feature extractor and feeding the output to another model specifically designed for the new task [Li et al., 2020]. In their ablation studies, [Devlin et al., 2018] compared the two fine-tuning strategies on a Named Entity Recognition (NER) task.

Given that the model used for the privacy attack in this thesis is a BERT-based NER model, NER will be discussed in greater detail here. Named Entity Recognition is a NLP task that involves identifying and classifying named entities in a given text into pre-defined categories (or labels) [Qiu et al., 2020]. NER is typically implemented within a sequence labeling framework and plays a crucial role in natural language understanding and information retrieval tasks by enabling systems to extract and understand relevant entities in a text [Qiu et al., 2020]. A Named Entity (NE) refers to a word or phrase that unambiguously identifies one item from a group of items that possess similar characteristics [Li et al., 2022]. This definition overlaps with the description of Personally Identifiable Information (PII)s, or more precisely direct identifiers, presented in a previous section about sensitive information. Examples of NEs can be of generic nature, like names, organizations, locations etc., or domain-specific, such as genes in the field of biology or bank account numbers in the financial sector [Li et al., 2022]. The sentence below in (1) illustrates a simplified example including NEs with their corresponding labels in brackets.

(1) <LOC On Time Square> <PER Anne> <PER Shirley> ate a <ORG Fairtrade> chocolate.

In (1) there are four different NEs (Time Square, Anne, Shirley, Fairtrade) of three different entity types (`location <LOC>, person <PER>, organization <ORG>`). A common approach in language modeling involves analyzing the text and then predicting the most

likely entity type for each word or phrase considering all possible entity types [Li et al., 2022]. So, in the case of (1), a neural tagger would calculate the probability of, for example, the identified named entity *Anne* being an instance of types `<LOC>`, `<PER>` or `<ORG>`. This probability distribution over all possible types is then used to assign the type with the highest probability `<PER>` to the named entity *Anne*.

Referring back to the comparison of the fine-tuning and feature-based strategies for NER-specific BERT, [Devlin et al., 2018] applied BERT to the CoNLL-2003 NER task[9] [Sang and Meulder, 2003] using both approaches. For fine-tuning, a classification layer is added to the pre-trained model, and all parameters are then adjusted collectively for the NER task. In contrast, for the feature-based approach, activations[10] are obtained from one or more layers of BERT without any adjustments to its parameters to use them as features for further processing. Based on the performance of both approaches on the NER task summarized in Table 2.2, [Devlin et al., 2018] deduce that pre-trained BERT can successfully be employed with both approaches. The performance metric presented in Table 2.2, i.e. the f1-score for the development dataset, is the harmonic mean of the two metrics precision and recall. Precision is a reliability measure that computes the proportion of True Positive (TP) to all positive results (TP and FP) [Goutte and Gaussier, 2005]. Recall is an effectiveness measure that calculates the proportion of TP to all actual positive results (TP and FN) [Sokolova and Lapalme, 2009]. To all three metrics applies that they range between 0 and 1, where 1 is the best achievable score and 0 the worst.

System	DEV F1 score
Fine-tuning	
BERT$_{Base}$	96.4
BERT$_{Large}$	96.6
Feature-based	
BERT$_{Base}$ + Concat Last Four Hidden Layers	96.1

Table 2.2.: Comparison of BERT results on the CoNLL-2003 NER task comparing fine-tuning and feature-based approach. Shortened version of Table 7 in [Devlin et al., 2018].

To summarize, LLM pre-training and fine-tuning have proven to be exceedingly successful in enhancing the performance of downstream tasks such as, e.g. Named Entity Recognition [Jurafsky and Martin, 2023]. LLMs can gain generic linguistic knowledge by

[9] A more in-depth analysis of the CoNLL-2003 dataset will be presented in the next chapter.

[10] Activations in this context are the output of one or more layers of the BERT model when processesing input data. These activations, often referred to as"feature maps", capture intermediate representations of the input data at various levels of abstraction [Sharma et al., 2020].

pre-training on large text corpora, which is then fine-tuned on task-specific datasets to achieve improved results on task-specific applications. However, with the use of vast amounts of unlabeled training data comes a growing concern over data quality and safety. For example, it is possible that personal or sensitive information may be included in LLM training datasets without the knowledge or consent of individuals involved. The next section will explore ethical implications of using potentially sensitive data in LLM training.

2.2.3. Training Data

So far, the main emphasis in this part has been on the training objectives and methods of LLMs. However, it is equally important to also consider the data used in the training process. Training data refers to the set of samples, typically represented as input-output pairs, used to train a model [Dargan et al., 2020]. In NLP, different types of training data are used for various tasks such as language understanding, generation, and translation. For example, training data for named entity recognition tasks typically consists of sentences with corresponding entity labels, while training data for machine translation tasks typically consists of parallel sentences in different languages.

The quality and quantity of training data is critical to the performance of the model, i.e. a large amount of high-quality data is needed for the model to generalize well and avoid overfitting[11][Deng and Liu, 2018]. Frequently, the source of datasets used to train LLMs are publicly available texts from the internet, such as Wikipedia, news articles, scientific papers, and books or other web-mined content [Kreutzer et al., 2022]. As mentioned previously, LLMs are usually trained on large datasets, which as [Bender et al., 2021] point out increases the difficulty of knowing what is in the training data. This issue also applies to the (unintended) inclusion of sensitive information of any form in the training data, which can be problematic as the risk of exposing sensitive data only arises from the presence of this data in the training dataset [Weidinger et al., 2021]. Another noteworthy aspect of web-crawled training datasets is that they could function as unintended archives for data that has been removed from the internet after the crawling process [Carlini et al., 2020]. These “archives” could then not only include sensitive information, but also harmful

[11] Overfitting is a common problem in machine learning. [Lever et al., 2016] provide a broad explanation of model over- and underfitting, which can be summarized as follows: model fit can be evaluated by comparing its predictions to new data (prediction error) or estimated and true parameter values (estimation error). Both types of errors are influenced by bias (error caused by using a model that is not capable of capturing the underlying model) and variance (error caused by sensitivity to noise in the data). A model that is overly complex will have low bias and high variance (overfitting), while a model that is too simple will have high bias and low variance (underfitting).

language and/or misinformation extracted from banned websites as [Gehman et al., 2020] observe.

The issue of data prevalence and quality has been recognized by the research community, which have identified that web-crawled datasets tend to be noisy[12], resulting in a lower quality of data compared to manually curated datasets [Kreutzer et al., 2022]. However, gathering and curating such massive datasets is an expensive and time-consuming process and might not be feasible in most cases. Efforts to develop a standardized method to document and annotate datasets used for training models have been suggested by [Gebru et al., 2021] universally for machine learning and for NLP systems in specific by [Bender and Friedman, 2018]. The latter do not allude to the topic of sensitive data, in the sense of Personally Identifiable Information, but rather concentrate on identifying and addressing biases in systems trained on naturally occurring language data. In [Gebru et al., 2021]'s proposal that datasets should be released with accompanying datasheets informing about the contexts and contents of the dataset, the handling of sensitive information is only mentioned peripherally. They do not go into details, but suggest to include a description of the relevant data if a dataset contains sensitive information. The reason for attaching little significance to sensitive data in these guidelines may be due to the fact that the mentioned approaches are primarily dedicated to promote transparency and fairness. Nevertheless, the collection and choice of training data instances may also influence the risk of involuntarily sharing sensitive information, but currently there is a lack of universally accepted industry-wide standards or implementations for administering sensitive data in training datasets. Despite this, it is critical to be aware of possible risks and dangers associated with accessing information through privacy attacks. The next section will explore this issue in more detail, providing an overview of the various approaches used to measure the risk of information leakage in LLMs.

2.2.4. Privacy Attacks on LLMs

As Large Language Models continue to advance, they become integrated into more and more different applications and technologies. Nonetheless, the growing use of LLMs has raised serious concerns about the dangers of privacy attacks. In the field of NLP, the term privacy attack has been applied to unauthorized or illicit attempts to extract sensitive information of individuals or groups from LLMs [Dwork et al., 2017, Rigaki and Garcia, 2020].

[12] Noisy data refers to data that contains errors, inconsistencies, or irrelevant information. The term can also be applied to data that includes irrelevant or redundant features that do not contribute to the problem being solved. [Wang et al., 1995]

According to a definition provided by [Nasr et al., 2019], privacy-sensitive information in this context refers to information about a model's training data, architecture or (hyper-)parameters that an adversary cannot derive from other models trained on similar data or in a comparable way. In the training data space, it can be further distinguished between information about the general data population and information related to individual training samples, with the latter being vulnerable to privacy breaches [Nasr et al., 2019]. So the target of privacy attacks are either the training data of models or the model itself. Depending on their objective and applied strategy, privacy attacks can be divided into four broad categories of attacks:

- **Membership Inference Attacks** aim at determining whether a specific record was used in the training of a machine learning model, by analyzing the output of the model for that record. This type of attack was first introduced by [Shokri et al., 2017], which has inspired further research in this field. Membership Inference attacks are developed under the supposition that a data record's membership in a model's training data would generally reflect in higher predictions scores, whereas unseen data records would yield lower scores in comparison [Hintersdorf et al., 2021].
- **Reconstruction Attacks** attempt to recreate training samples and/or their non-sensitive training labels. This type of attack has also been referred to as *attribute inference* [Yeom et al., 2018] and *model inversion* [Salem et al., 2019] since it involves utilizing the output of a machine learning model to infer sensitive features or retrieve the entire data sample.
- **Property Inference Attacks** are designed to extract information that was learned by the model unintentionally and that is not relevant to the model's primary training task [Rigaki and Garcia, 2020]. One example of this type of attack is the memorization attack, which seeks to uncover sensitive patterns in the training data of the target model [Carlini et al., 2018].
- **Model Stealing Attacks** intend to obtain the parameters [Tramèr et al., 2016] or hyperparameters [Oh et al., 2017] of a target machine learning model. In many cases trained models count as intellectual property and are regarded as confidential, which renders the retrieval of models' (hyper-) parameters privacy violations [Mireshghallah et al., 2020].

All of the above-mentioned privacy attack types can be further specified. An overview of general properties of privacy attacks is presented in Table 2.3. Initially, there is the

question of which kind of access quality is given for the respective attack scenario, because the quality of the access determines the nature of input for the attack [Nasr et al., 2019]. This access can either be *black-box* or *white-box*. In a white-box attack the adversary has knowledge about the model's parameters, architecture or training data [Rigaki and Garcia, 2020]. Attacks in which the adversary's access is more limited are called black-box attacks. In those attacks, adversaries can only use the model's output to their queries to gain information [Liu et al., 2021]. Another important factor is the mode of the attack, which provides details about the timing of the attack. *Active* attacks assume that the adversary is part of the training process, as in federated learning[13] and can thus actively manipulate the target model [Nasr et al., 2019]. *Passive* attacks typically occur after the training process is completed and do not interfere with the learning process of the model [Nasr et al., 2019]. Another significant element of an attack is the adversary's prior knowledge. An adversary with *supervised* knowledge uses a dataset, which comprises a subset of the training data from the targeted model to train an attack model to discover insights from the target model [Nasr et al., 2019]. An adversary with *unsupervised* knowledge, on the other hand, cannot exploit a labeled dataset for their attack, but has to find a structure or patterns without conducive identifiers [Nasr et al., 2019].

Attribute	Type	Description
Access	black-box	only query access to the target model, without knowledge on the model's architecture or (hyper-)parameters
	white-box	access to full target model, including architecture and (hyper-) parameters
Mode	passive	attack (mostly) during inference; does not interfere with the training phase of a model
	active	attack during the training phase of the model to interfere with the learning process
Knowledge	supervised	the adversary possesses a dataset $D\prime$ which overlaps with the target dataset D.
	unsupervised	the adversary has no information about whether a data sample is in the target dataset D.

Table 2.3.: Taxonomic overview of general properties to define privacy attacks on LLMs - adapted version of Table 1 in [Nasr et al., 2019].

[13] Federated learning is a form of distributed machine learning in which multiple data owners contribute data to collaboratively train a model by only sharing parameter updates from local model training [McMahan et al., 2017].

The list given in Table 2.3 is not exclusive, additional properties that can describe the nature of a privacy attack exist, but within the scope of this thesis, access, mode and knowledge level of an attack are the most fundamental descriptive properties. It should also be noted here, that this thesis deals with *indirect* information exposure, which means attack scenarios in which the adversary attempts to infer model information but does not have access to this information [Mireshghallah et al., 2020]. In contrast to *direct* information leakage, in which data breaches are precipitated by acquiring access to the actual model information [Mireshghallah et al., 2020].

Independent from the attack type, one of the most widely-researched type of models in this context are generative models, which are trained to produce textual or visual output ([Shokri et al., 2017, Song et al., 2017, Yeom et al., 2018, Carlini et al., 2018], among others). For example, in a number of experiments with GPT-variants investigating the disclosure of unique or rare sequences from the training data of text-generating LLMs, [Carlini et al., 2018, 2020, 2022] found that model scale, data duplication in the training data and input context are among the most influential factors for training data memorization/extraction in neural networks. Regarding the connection of model scale and data leakage, [Tirumala et al., 2022] affirm that larger models tend to memorize more training data. They further highlight that specifically nouns, proper nouns[14], and numerals are memorized remarkably faster than other word classes.

A smaller number of investigations address the issue of information leakage in sequence-labeling models [Qiu et al., 2020, Lehman et al., 2021, Vakili and Dalianis, 2021]. Based on the difference in task, these approaches primarily use BERT-based models as targets for their attacks. For example, [Lehman et al., 2021] and [Vakili and Dalianis, 2021] focus on BERT versions trained on digital health records in an attempt to check if the targeted model is at risk of leaking sensitive information. They conclude that the likelihood of successfully extracting sensitive information from a BERT-based model is significantly lower when compared to a GPT model. To broaden the current knowledge about data leakage in BERT-based models and to assess the influence of model size, data duplication and probing context on privacy breaches for neural tagger models, this thesis will propose an analysis framework to investigate the risk of identifying named entities, to be specific PER names, as members of the training dataset.

[14] A proper noun is a specific name for a person, place, or thing [Brown and Miller, 2013].

3. Methodology

This section describes the methodology used to investigate sensitive information leakage in a BERT-based Named Entity Recognition model. First, an overview of the targeted NER model, the datasets and the input prompts used in the privacy attack is given. Then the experimental setting for the simulated attack is explained in detail. The presented analysis framework follows an exploratory approach to thoroughly assess the leakage risk of a NER model in a narrowly defined attack scenario. Additionally, the framework offers a deeper understanding of the potential privacy risks inherent in NER models.

3.1. Model Description

This research examines a BERT-based NER model, which comes in two sizes, the *bert-base-NER* and the *bert-large-NER* model [Lim, 2020a, Lim, 2020b]. For simplicity reasons, *bert-NER* will be used as an umbrella term to refer to both model sizes. Bert-NER is a fine-tuned, cased BERT model trained on a Named Entity Recognition task and provided open-source by Hugging Face. The bert-NER model was chosen because it is the second[1] most downloaded NER model on Hugging Face (status January, 2023) and the most downloaded NER model trained on an English dataset (950,000 downloads, both model sizes taken together). The training dataset of bert-NER constituted a second reason to select this model for a detailed analysis. Bert-NER was trained on the CoNLL-2003 dataset [Sang and Meulder, 2003], a widely used dataset for NER tasks. The CoNLL-2003 dataset will be described in more detail in section 3.2.1.

For the training of the bert-NER model, the authors used the recommended hyperparameter values for batch size, learning rate and number of epochs from the original BERT paper. Bert-NER uses a subword-based tokenizer [PreTrainedTokenizerFast, version 4.26.0,], which depends on the transformer's *tokenizer* library. Subword-based tokenization operates on the principle that frequent words should not be split into smaller units, whereas

[1] The most downloaded NER model on Hugging Face at the time of writing this thesis was: camembert-ner trained on a French dataset (2,150,000 downloads as of January, 2023).

rare words should be decomposed into meaningful sub-units [Sennrich et al., 2015]. If we recall the example sentence (1) from section 2.2.2, a subword tokenizer algorithm renders (1) into (2).

(2) `['On', 'Time', 'Square', 'Anne', 'Shirley', 'ate', 'a', 'Fair', '##tra', '##de', 'chocolate', '.']`

The two octothorpe symbols "##", as in '##tra' or '##de' in Example (2), inidcate that these tokens are completions of the previous one and can be attached to the precedent token without space (i.e. 'Fair' + '##tra' + '##de' = 'Fairtrade').

As can be seen in Table 3.1 the fine-tuned bert-NER model closely follows the pre-trained baseline model regarding number of layers and parameter count.

Model	Type	Training objective	Layers	Parameters
BERT_{Base}	baseline	MLM + NSP	12	110 Mio.
BERT_{Large}	baseline	MLM + NSP	24	340 Mio.
bert-base-NER	fine-tuned	NER	12	108 Mio.
bert-large-NER	fine-tuned	NER	24 hidden + 3 FC layers	333 Mio.

Table 3.1.: Comparison of baseline and fine-tuned BERT models.

The training objective of the bert-NER model consisted of assigning pre-defined NE-tags to tokens from the labeled CoNLL-2003 dataset. In contrast to generative models such as GPT-variants which produce textual output, a NER model outputs a prediction of the most likely NE-tag for the inserted token. In theoretical terms, the final layer of a NER model yields a probability distribution over the different entity tags, i.e. the prediction scores for each entity tag, by using an activation function, in the case of bert-NER the SoftMax function [Goodfellow et al., 2016]. Therefore, the prediction scores in this model range between 0 and 1 for each entity tag, whereas the prediction scores of all entity tags have to sum up to 1. With an interposed post-processing step, the model reports only the entity tag with the highest prediction score for the respective token. In some studies, the authors refer to these prediction scores also as confidence values or scores, but with regard to research in the area of explainable AI [Samek et al., 2017, van der Waa et al., 2020] arguing that measures like the SoftMax, which represent an estimated likelihood for a certain prediction label, often lack transparency and explainability, this terminology will not be used in this thesis.

Considering the tokenized example sentence (2), a bert-base-NER output for the token `'Anne'` is illustrated in Example (3). The model classified `'Anne'` as a `'B-PER'` entity[2]

[2] A detailed description of the various entity tags is provided in section 3.2.1.

with a normalized prediction score of 0.9993299.

(3) `{'entity':'B-PER', 'score':0.9993299, 'index':4, 'word':'Anne'}`

Regarding the performance of bert-NER, the authors report good results (cf. Table 3.2), although they are slightly lower than the original BERT results summarized in Table 2.2. A definition for the evaluation metrics, f1-score, precision and recall is given in section 2.2.2.

bert-base-NER		
Metric	DEV set	TEST set
f1-score	95.1	91.3
precision	95.0	90.7
recall	95.3	91.9
bert-large-NER		
Metric	DEV set	TEST set
f1-score	95.7	91.7
precision	95.3	91.2
recall	96.1	92.3

Table 3.2.: Evaluation of the performance of BERT-based NER models. Results are taken from [Lim, 2020a, Lim, 2020b]

3.2. Data Description

Based on the problem definition in Section 1.2, an attack dataset is created to investigate the (non-)membership of data samples in the targeted model's training data. The attack dataset used in this analysis is balanced, meaning that it contains an equal number of positive samples pos_i, i.e. PER name in training data, and negative samples neg_i, i.e. PER name not in training data. This balance helps to ensure that the results obtained from this dataset are more generalizable and applicable to a wider range of scenarios.

The attack dataset can be organized in two different ways, using the same set of data instances, but with different data configurations. Dataset A is arranged as a set of randomly aggregated n pairs of positive and negative samples (pos_i, neg_i), while dataset B is structured as a list of positive and negative samples. By varying the internal structure of the attack dataset, various attack strategies can be tested to estimate the risk of sensitive data leakage in bert-NER. This section describes the data sources, the steps of data preparation and the input prompt corpus for the privacy attack on bert-NER.

3.2.1. Positive Samples: Model Training Dataset

The bert-NER model was fine-tuned on the English version of the standard CoNLL-2003 Named Entity Recognition dataset [Sang and Meulder, 2003]. The English CoNLL-2003 dataset is a collection of news wire articles from the Reuters Corpus [Lewis et al., 2004] from mid 1996 to mid 1997. The corpus contains over 800,000 articles in total, which were written in English and cover a wide range of topics including business, politics, and sports. CoNLL-2003 is considered a high-quality dataset with a large number of examples of named entities from various domains, making it a valuable resource for NER research [Derczynski et al., 2016].

The CoNLL-2003 dataset is split into training, development and test data. Together these three datasets contain over 300,000 tokens with entities labeled according to four types: person `PER`, organization `ORG`, location `LOC`, and miscellaneous `MISC`. Table 3.3 presents an overview of the various tags used in the CoNLL-2003 dataset. To identify spans of tokens that belong to the same named entity, BIO-tagging [Ramshaw and Marcus, 1995] is used in addition to the named entity tagging. Tokens tagged with `O` are outside of any named entity span, the `I-XXX` and `B-XXX` tag are used to indicate whether two immediately consecutive tokens of the same entity type belong to the same span, with `B-XXX` marking the beginning of a new independent span [Sang and Meulder, 2003].

NE-tag	Description
`O`	outside of NE
`I-PER`	person's name
`B-PER`	[Beginning] of a person's name immediately following an `I-PER` entity
`I-ORG`	organization
`B-ORG`	[Beginning] of an organization immediately following an `I-ORG` entity
`I-LOC`	location
`B-LOC`	[Beginning] of a location immediately following an `I-LOC` entity
`I-MISC`	miscellaneous entity
`B-MISC`	[Beginning] of a miscellaneous entity immediately following an `I-MISC` entity

Table 3.3.: Overview of NE-tags used in CoNLL-2003 dataset.

Regarding the data format of the CoNLL-2003 dataset, the dataset has one token per line, and four fields per line. The first item on each line is a token, the second a Part of Speech (POS) tag, the third a syntactic chunk tag and the fourth a named entity tag from the list given in Table 3.3. Some sample instances contained in the dataset are provided in

Table 3.4.

token	POS-tag	chunk tag	NE-tag
Meidlinger	NNP	I-NP	I-PER
the	DT	B-NP	O
World	NNP	I-NP	I-MISC

Table 3.4.: Sample of bert-NER training data

For the attack, only the training dataset of the CoNLL-2003 corpus is relevant as this is the same data used in the training process of bert-NER. This dataset includes 6600 instances of PER entities, i.e. tokens with NE-tag `I-PER` or `B-PER`. However, this number comprises duplicates and single token PER names, which are redundant or undesirable data samples for the simulated attack. Therefore these instances were removed in a data cleaning process to obtain a high-quality dataset holding 2,645 unique, multi-token[3] PER names.

3.2.2. Negative Samples: Wikidata Name List

The source for the negative samples, i.e. PER names that are not included in the training dataset of bert-NER, is the knowledge base Wikidata [Vrandecic and Krötzsch, 2014]. Wikidata is an open-source, linked data platform that allows users to create, manage, and use data in a structured format [Färber et al., 2017]. Additionally, Wikidata supports rich querying capabilities, which allow to retrieve and aggregate data from the knowledge base. Thus, the rationale behind using Wikidata as a source for the negative samples was to quickly and easily compile a list of actual names of real-world persons. Factors that would also be relevant for potential adversaries with the intention to create large PER name lists in order to attack NER models.

To access data stored in Wikidata, a SPARQL query has to be formulated defining the relevant data objects. With the query in 3.1, entities of the instance human (`wd:Q5`) are selected and their names (`personLabel`) are returned. By using a relatively unconstrained query like 3.1, Wikidata returns a very high number of person records.

```
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX schema: <http://schema.org/>
```

[3] No upper limit was set for the accumulation of successive `I-PER` tokens. In practice, the maximum number of tokens for one PER name was reached with five consecutive tokens.

```
SELECT DISTINCT ?person ?personLabel
WHERE {
  ?person wdt:P31 wd:Q5 .
  ?person schema:name ?personLabel .
  FILTER (lang(?personLabel) = "en") .
}
```

Listing 3.1: Query to Knowledge Base Wikidata

However, only person names that include characters supported by the CP-1252 encoding are considered for the experiments in this thesis. The character set CP-1252 is the default encoding for text files in Windows-based systems and is used for the encoding of English and most European languages [Code page, 1252]. The reason for this restriction is that the bert-NER training data is encoded in CP-1252. Therefore, their characters should be recognizable for the model in order to guarantee a fair comparison between the sampled PER names. Because of this restriction, 292,426 PER names retrieved from Wikidata could not be included. In all other aspects of data cleaning, the same constraints are applied to the names from the Wikidata dataset and the names contained in D_{train}. During the data cleaning process, duplicates and single-token instances of PER names were filtered out. Finally, 7,617,797 multi-token PER names form the dataset W, which serves as a sampling pool for the creation of a dataset $W\prime$, a subset of W without D_{train}. In other words, $W\prime$ does not contain PER names found in D_{train}. The next section outlines how the positive samples dataset derived from D_{train} and the negative samples dataset $W\prime$ obtained from Wikidata are merged to build a dataset for the attack on the targeted model bert-NER.

3.2.3. Attack Dataset

The illustration shown in Figure 3.1 represents a diagrammatic overview of the creation process for the attack datasets A_{dev} and A_{test}. The source for the positive samples pos_i is the bert-NER training dataset D_{train} and for the negative samples neg_i the source is a Wikidata name list W as explained in the previous sections. In the next step, the datasets $D\prime$, a subset of D_{train}, and $W\prime$, a subset of W, are generated.

$D\prime$ consists of all sample PER names that are in the intersection of D_{train} and W. The magnitude of $D\prime$ expressed in numbers can be seen in Table 3.5. As the condition $|D\prime| = |W\prime|$, where $|D\prime|$ denotes the size of the intersection D_{train} and W, must hold true to be able to compose positive - negative sample pairs for a pair-wise approach in the experiments, the number of randomly sampled names from W is based on the number of

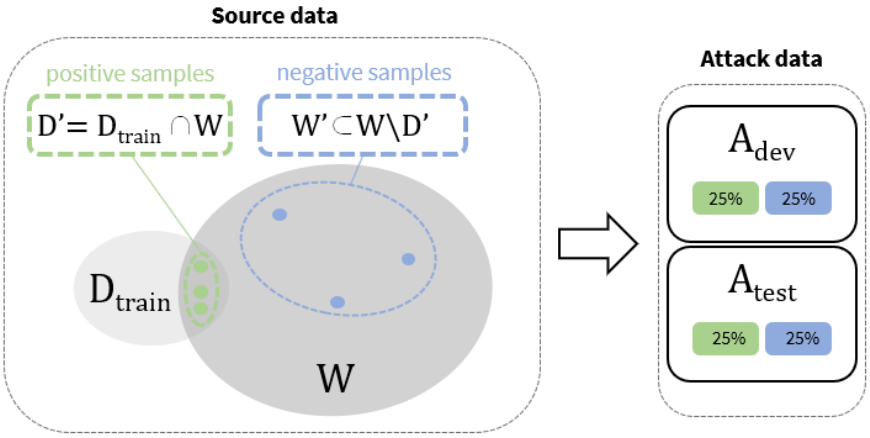

Figure 3.1.: Illustration of creation of attack datasets A_{dev} and A_{test}.

positive samples in $D\prime$. To create dataset $W\prime$, n (= $|D\prime|$ = 1653) PER names are randomly selected from W, provided that the samples meet the condition $\{neg_i \in W \setminus D\prime\}$. It is essential to highlight here, that the two sets $D\prime$ and $W\prime$ are completely disjoint, i.e., $D\prime \cap W\prime = \varnothing$. Each sample name can only occur once in the attack dataset A, so that for any elements x and y in A, $x \neq y$ applies. Subsequently, the attack dataset A consists of 50% positive and 50% negative samples, which build the foundation for the a) the pair-wise approach and b) full-list approach of the experiments. For the latter, the internal structure of A does not need to be changed further.

To combine the sample names from the datasets $D\prime$ and $W\prime$ into positive-negative sample pairs for the pair-wise approach, an additional step has to be taken. Theoretically this process can be described as follows: Let pos_i be a data record from the model's training dataset $D\prime$ and neg_i be a randomly sampled name from the dataset $W\prime$, then pos_i and neg_i will form a unique pair (pos_i, neg_i) in dataset A. Symbolically this can be written as:

$$pos_i \in D\prime, neg_i \in W\prime \rightarrow \exists!(pos_i, neg_i) \in A \tag{3.1}$$

Additionally, the elements of the sample pairs (pos_i, neg_i) in A are assigned a binary label (0/1) indicating if they belong to the training dataset of bert-NER, i.e. $D\prime$. 1 denotes positive samples from $D\prime$ and 0 signifies the absence from $D\prime$.

In order to have one dataset to measure the performance of different attack strategies and one dataset to verify the performance of the attack, the attack dataset A is subdivided

with a ratio of 50:50 into a development A_{dev} and test dataset A_{test}. Table 3.5 summarizes the PER name count in the individual datasets.

source data	# of PER names
Wikidata W	7,617,797
bert-NER D_{train}	2,645
attack datasets	# of PER names
$D\prime$	1,653
$W\prime$	1,653
A_{dev}	826
A_{test}	827

Table 3.5.: Number of PER names per dataset. The numbers here reflect the data after a data cleaning process that considered only multi-token names with characters encoded in CP-1252 and involved removing duplicate names.

It shall be noted here that the proportion of training data PER names contained in the Wikidata name list W is 0.0002%, whereas 62.5% of the training dataset PER names can be found in W. This means that a high percentage of PER names in D_{train} can be covered by a simple Wikidata query. The advantage of easy access to a large amount of possibly sensitive information may be exploited by adversaries who can use name lists like W to initiate privacy attacks on NER models.

The PER names in the attack dataset A are presented to the bert-NER model without context as well as part of longer prompt sequences. To check whether the contextual surrounding influences the likelihood of detecting training data samples, various input prompts are devised to embed the probed names from dataset A. The next section 3.2.4 presents the input prompts used in different attack strategies which are outlined in section 3.3.

3.2.4. Input Prompts

Input prompts are the text input provided to a language model. Given an input prompt, the model generates a task-specific response. In the case of bert-NER, named entities in the input prompt are identified and classified according to the pre-defined NE-types (cf. Table 3.3). For the simulated attack, nine different input prompts are composed. Table 3.6 lists the nine input prompt contexts $c_1, ..., c_9$ together with their linguistic features.

The PER names from the attack datasets A_{dev} and A_{test} are individually inserted at the position of a placeholder [MASK] token (= t_{mask}) into the different input prompt contexts

c_i and fed as a complete sequence to bert-NER. The respective sequences of the input prompt corpus C vary in length, type of utterance and syntactic position of the PER name tokens, but $t_{mask} \in c_i$ always holds true.

i	Input Prompt c	Type of Utterance	Syntactic Position of PER token
1	[MASK]	n/a	n/a
2	[MASK] is a person.	assertion	subject
3	My name is [MASK].	assertion	object
4	I am named [MASK].	assertion	object
5	[MASK] is an individual human being.	assertion	subject
6	Is [MASK] your name?	interrogative	subject
7	Is [MASK] a person?	interrogative	subject
8	[MASK] is not a person.	negation	subject
9	My name is not [MASK].	negation	object

Table 3.6.: Input prompts for querying the model. The [MASK] token is replaced with the respective PER name being queried.

Input prompt c_1 only contains the respective sample name s_i without additional context. The rest of the input prompts c_2 - c_9 are designed to embed the sample names in semantic proximity to the concept of Personally Identifiable Information. For example, input prompt c_5 is a dictionary definition for the countable noun “person” [Collins Dictionary,]. By varying the syntactic position of the inserted sample names and the type of utterance possible effects on the data leakage potential of accompanying context can be examined. The next section 3.3 describes the use of the input prompts and the implementation of different attack strategies.

3.3. Experimental Setting

After configuring the attack datasets A_{dev} and A_{test} and composing input prompts for bert-NER, the next step involves defining the analytical framework to evaluate the sensitive data leakage risk from the targeted model. For this, let us recall the three characters introduced in the problem statement section 1.2. The aim of Alex, Blake and Charlie is to simulate an attack on the bert-NER model to estimate the risk of sensitive information leakage in this model. They all have different responsibilities in this simulated attack; Alex, as the model and data owner, knows the labels [0/1] of the sample names (pos_i, neg_i) in the attack datasets and can therefore evaluate the attacks performed by Blake and Charlie.

Blake and Charlie apply different attack strategies, although Charlie's attack only serves as a baseline for Blake's attacks.

Considering the categorization of privacy attacks along the lines of the criteria formulated in Table 2.3, the attack by Alex, Blake and Charlie on bert-NER can be classified as **black-box, passive and supervised**. The access supplied to the adversaries Blake and Charlie is specified as black-box because they do not have direct access to the model or the training data, but only query access. However, Alex as the model and data owner provides supervised knowledge regarding the membership of the probed sample names in bert-NER's training data. The mode of the attack is defined as passive because the model is attacked after the training process is completed.

The analytical framework for the attack is based on the same assumption as membership inference attacks: if a data record was used in the training of a model, the model will assign a higher prediction score to it compared to a novel record [Mireshghallah et al., 2020]. According to [Nasr et al., 2019], the last layer of a language model, i.e. the model output, discloses the most information about a data record's potential membership in the training data[4]. Using the model output as the main indicator for determining the risk of information exposure is a practical choice because the interpretation of output prediction scores is less ambiguous than the interpretation of, e.g. gradient values and from a pragmatic perspective, the output is more readily accessible for adversaries.

The attack process can be divided into two steps. In the first step, the bert-NER model is prompted with the prepared attack dataset A and inputs from corpus C. In the second step, the attack strategies of adversary Blake are applied to the results gained from step 1. To assess the informative value of Blake's attack strategies, they are compared to Charlie's baseline attack approach.

Step 1

The privacy attack uses the balanced set of of positive and negative sample names from the attack dataset $A = \{(pos_i, neg_i)\}_{i=1}^{n}$, where pos_i denotes the full name from the positive sample and neg_i denotes the full name from the negative sample of the i-th pair, in the case of the pair-wise approach. n denotes the size of the respective attack dataset, A_{dev} or A_{test}. Given a corpus of pre-defined input prompts $C = \{c_1, ..., c_9\}$ (cf. Table 3.6), the individual sample names are inserted in place of the [MASK] token into the input

[4] For the sake of completeness, it should be noted that [Nasr et al., 2019] further point out that the information gain regarding training data membership might be greater with gradients compared to output layers. However, the attack approach in this thesis concentrates on the model output.

sequences of C and passed as input through the model. When the target model is queried with the supplied inputs, the model returns NER-tag predictions and the corresponding prediction scores for the input tokens.

The focus in this framework lies on the prediction (scores) of PER-tags assigned to the probed sample names by the bert-NER model. As the probed sample names consist of multiple tokens, each individual token receives a prediction score for the PER-tag. To make the PER-tag prediction scores more comparable among sample names with different token numbers, the tokens' PER-tag prediction scores are averaged for each sample name. An alternative solution, though less exhaustive, would have been to take the minimum or the maximum score but both approaches would have given too much weight to one token of a multi-token entity. This can be illustrated briefly with the name "Anne Shirley" already known from Examples (1) - (3). The score for the token "Anne" is 0.9993299 and for the token "Shirley" 0.999025, for the full sample name "Anne Shirley" this constitutes an averaged prediction score of 0.99917745[5]. These averaged prediction scores per sample name, μ_{pos_i} for positive samples and μ_{neg_i} for negative samples, are further exploited as the main source of information whether a data sample is a member of the model's training dataset. Figure 3.2 provides a schematic overview of the proceedings in step 1 of the simulated attack on bert-NER.

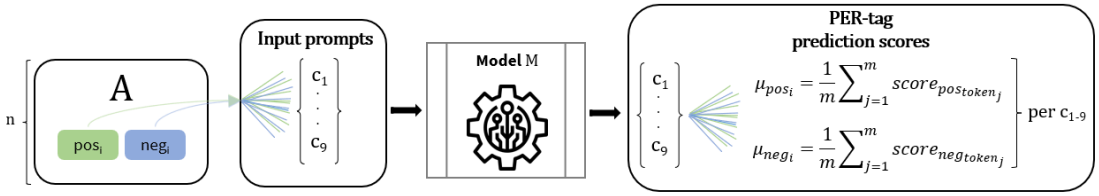

Figure 3.2.: Schematic view of **Step 1** in attack workflow

Step 2

The post-processed model output, i.e. averaged prediction scores for positive and negative samples (μ_{pos_i}, μ_{neg_i}) for each input prompt ($c_1, \ldots, c_9$), constitutes the starting point for the second step of the analytical attack framework. A simplified representation of the approach and the involved roles is shown in Figure 3.3.

The adversary Blake follows the hypothesis that neural taggers like bert-NER assign higher prediction scores to tokens known from the training process than to unseen to-

[5]The prediction score was not rounded for the sample names to not distort the results.

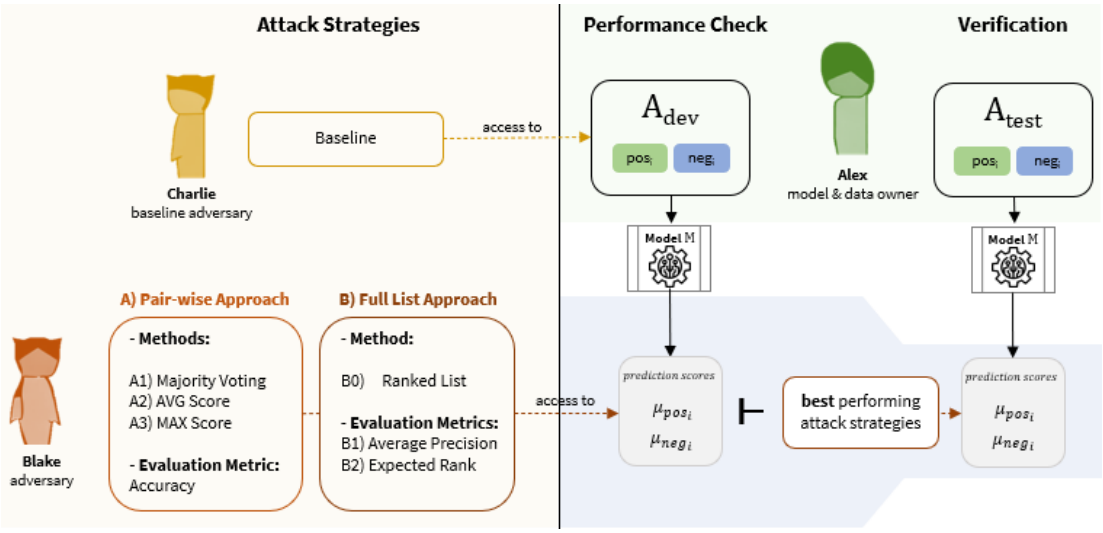

Figure 3.3.: Schematic view of **Step 2** in attack workflow

kens. Therefore, Blake develops different attack strategies for both the (A) pair-wise attack dataset and (B) the full-list attack dataset, leveraging the same sample name prediction score values. The purpose of these attack strategies is to estimate the risk of PER name leakage in bert-NER. In addition to Blake's strategies, Charlie defines a simple baseline strategy for (A) and (B) to validate the risk evaluation of Blake's attack strategies in comparison to pure guessing. For the baseline strategy the prediction score is not relevant. The performance of the different strategies is gauged by Alex, who has knowledge about the origin ($D\prime$ or $W\prime$) of the sample names in attack datasets A_{dev} and A_{test}.

(A) Pair-wise Approach

(A1) Majority Voting: Majority voting is a simple ensemble method used in machine learning and data science for combining multiple decisions of different classifiers into a single, consensus prediction [Lam and Suen, 1997]. The principle underlying majority voting utilizes the advantage of multiple classifiers in order to obtain a more robust and accurate overall prediction [Dietterich, 2000]. In this research, the majority voting result is not based on multiple classifiers but on multiple input contexts, i.e. queries, to the model. This majority voting approach uses the prediction scores of sample pairs (pos_i, neg_i) for all nine input prompts to determine which pair member is also a training dataset member.

Given a set of prediction score pairs $\{(\mu_{pos1}, \mu_{neg1}), (\mu_{pos2}, \mu_{neg2}), ..., (\mu_{posj}, \mu_{negj})\}$,

where (μ_{posj}, μ_{negj}) is the prediction score for positive and negative sample pair names in input context j from corpus C, the majority voting prediction is defined as follows: $\mathsf{k}_j = \begin{cases} 1, \ \mu_{posj} > \mu_{negj} \\ 0, \ \mu_{negj} > \mu_{posj} \end{cases}$

$$Kp = \sum_{j \in C} k_j \tag{3.2}$$

$$Kn = |C| - Kp$$

The winner, $winner_i$, of pair (pos_i, neg_i) can be expressed as:

$$winner_i = (pos_i \text{ if } Kp > Kn, \ neg_i \text{ if } Kn > Kp)$$

In the formula presented in 3.2 a $winner_i$, i.e. sample pair name with higher prediction scores in more than $|C|/2$ cases, is determined by comparing the number of individual context wins (Kp, Kn) for pos_i and neg_i. To evaluate the majority voting method, the sample names labeled by the majority voting as "winners" are compared with the true training data names pos_i in the attack dataset A and the accuracy of the majority voting approach is calculated.

(A2) AVG Score: The average score strategy uses the prediction scores μ_j of the sample name s_i for all input prompts c_j and calculates the mean PER-tag prediction score for each element of the sample pair (pos_i, neg_i) over the entire input corpus C. The mean scores of the pair, $mean(\mu_{posi}), mean(\mu_{negi})$, are compared against each other and the sample name with the higher mean score is determined as training data instance of bert-NER.

$$mean(\mu_{s_i}) = (1/j) * (\mu_{s_1} + \mu_{s_2}, + ... + \mu_{s_j}) \tag{3.3}$$

The winner, $winner_i$, of pair (pos_i, neg_i) can be expressed as:

$$winner_i = \begin{cases} pos_i, \ mean(\mu_{posi}) > mean(\mu_{negi}) \\ neg_i, \ mean(\mu_{negi}) > mean(\mu_{posi}) \end{cases}$$

Similar to the majority voting approach, the results of the AVG score method are compared with the true training data instances of bert-NER and the performance measured by the accuracy of prediction.

(A3) **MAX Score:** The maximum score strategy uses the highest PER-tag prediction score of the sample name for the entire input prompt corpus C. The maximum score for pos_i and neg_i of pair$_i$ are compared against each other and the sample pair name with the highest maximum score is assumed to be a training data instance of the target model M. In case both pair sample names have an identical MAX PER-tag score, the sample name with the higher AVG score is considered $winner_i$.

$$winner_i = \begin{cases} pos_i,\ max(\mu_{posi}) > max(\mu_{negi}) \\ neg_i,\ max(\mu_{negi}) > max(\mu_{posi}) \end{cases} \tag{3.4}$$

The MAX Score method is another pair-wise attack strategy, for which the results are compared with the true training data instances of bert-NER. The accuracy of the prediction is then used to measure its performance.

(A4) **Baseline:** Given a sample pair (pos_i, neg_i), a dummy classifier randomly predicts with equal probability, which pair member is from the training dataset [1] and which pair member is not from the training dataset [0]. The code example in (3.2) shows the implementation of the dummy classifier in Python. Both classes are equally likely, similar to a coin toss decision. For adversary Blake's attack strategies to be successful, their accuracy must exceed the baseline value of approximately 0.5, which amounts to random guessing.

```
import random

class PairDummyClassifier:
    def __init__(self):
        self.classes_ = [[1, 0], [0, 1]]

    def predict(self, X):
        return random.choice(self.classes_)
```

Listing 3.2: Simple Dummy Classifier used as baseline to classify sample pair elements (pos_i, neg_i) at random with equal probability for being a member (1) or non-member (0) of D_{train}.

(B) **Full-list Approach** In addition to the pair-wise sampling approach, also the full list of sample names in the attack dataset A is used. In this approach the list of sample names is ranked by their individual PER-tag prediction score for the respective query μ_i.

(B1) Average Precision: Average Precision (AP) is commonly used as an evaluation metric for information retrieval tasks where the objective is to retrieve a set of relevant items from a large collection [Manning et al., 2008]. The relevance of each item is binary, either relevant or not relevant, which maps to the attack approach in this thesis, as sample names can also be either in D_{train} or not in D_{train}, or, in other words, can be a relevant training data instance or not.

$$AP = \sum_{i=1}^{n} (Rec_i - Rec_{i-1}) \cdot Prec_i \tag{3.5}$$

where:

n is the number of items in the ranked list.

Rec_i is the recall at rank i.

$Prec_i$ is the precision at rank i.

The Average Precision per query is calculated by ranking the prediction scores (μ_{pos_i}, μ_{neg_i}) for each context input and then applying the formula in 3.5 [Zhang and Zhang, 2009] to the ranked list. The result for each context input, i.e. query, is an AP value ranging from 0 to 1, where 1 denotes a perfect AP. To summarize the effectiveness of rankings from multiple queries, the arithmetic mean of the individually computed APs, termed the Mean Average Precision (MAP), can be calculated [Croft et al., 2009]. MAP is a frequently used effectiveness measure because it reflects the idea that top-ranked items are considered the most important [Croft et al., 2009] and it has been demonstrated that the MAP score has particularly good stability and discrimination [Manning et al., 2008].

(B2) Expected Rank: $E(rank|pos_i)$ expresses a conditional expectation given a randomly selected positive sample pos_i (= PER name from D_{train}) and its expected rank in a ranking of PER-tag scores. This can also be formulated as the expected number of guesses until pos_i is encountered in a list of sample names. The accuracy in the first summand of the formula presented in 3.6 is derived from the accuracy observed in the direct comparison of (μ_{pos_i}, μ_{neg_i}) and can be determined for each input prompt ($c_1, ..., c_9$) or as an average accuracy for the entire input corpus C. n_{neg} and n_{pos} denote the number of negative and positive samples, respectively. The second summand $[0.5\ n_{pos}]$ is a constant, which considers also other positive samples in the ranking.

$$E(rank|pos_i) = (1 - accuracy) \cdot n_{neg} + 0.5 \cdot n_{pos} \tag{3.6}$$

As the expected value of the rank in $E(rank|pos_i)$ is measured on a sample, i.e. attack dataset A, it presents a good estimate for this value. In contrast to the accuracy in the pair-wise attack strategies (A1-A3) and the average precision (B1), a low value ($\geq$1) is desired for the expected rank to demonstrate that the attack was successful[6].

(B3) **Baseline:** Given a balanced list of PER names, consisting of 50% training data instances and 50% non-training data instances, a dummy classifier assigns binary labels [0, 1] at random with an equal probability of 0 (not in training data) or 1 (in training data) to the names in the list. The accuracy in this case is to expected to be close to 0.5. (3.3) shows a code example of the implementation of the dummy classifier in Python.

```
import random
import numpy as np

class ListDummyClassifier:
    def __init__(self, random_seed=1653):
        self.random_seed = random_seed

    def fit(self, X):
        np.random.seed(self.random_seed)
        self.dummy_labels = np.random.choice([0, 1], size
    =len(X), p=[0.5, 0.5])

    def predict(self, X):
        return self.dummy_labels
```

Listing 3.3: Simple Dummy Classifier used as baseline to classify sample names in a list at random with equal probability for being a member (1) or non-member (0) of D_{train}.

The above-mentioned attack strategies for the leakage risk analysis in bert-NER are first optimized and evaluated on attack dataset A_{dev} in its (A) pair-wise and (B) full-list configuration. The performance of the various strategies in this process and practical

[6]From another point of view, a high value for the expected rank could be preferable implying that the model is not easily prone to leak sensitive information.

considerations will decide which attack strategy for the pair-wise (A1-A3) and full-list approach (B1-B2) will be tested on the second attack dataset A_{test}. The next chapter presents the results of the simulated attack outlined in this section.

4. Results

This chapter conducts an in-depth analysis of the investigation into sensitive information leakage in a BERT-based NER model. It provides general insights into the potential for leakages of named entities, i.e. Person (PER) names, from the model's training dataset and presents the findings of the simulated attack. The performance of the different attack strategies outlined in the previous chapter is used to estimate the risk of leakage in the examined language model. The chapter is divided into two parts: the first part discusses general findings of the attack approach based on the model's output and examines the hypothesis that training data samples tendentially receive higher prediction scores than non-training data samples. The second part concentrates on the findings of the simulated attack highlighting both the strengths and limitations of the presented strategies. The results of this research attempt to contribute to a better understanding of information leakage in Large Language Models by providing a collection of strategies within a risk assessment framework to gauge the scope and contingency of training data leakages.

4.1. General Findings

This section provides a comprehensive overview of general findings with a particular focus on addressing the question whether the prediction score of PER-tags represents a reliable indicator for determining whether a sample name was included in the training dataset of bert-NER.

Tables 4.1 and 4.2 summarize the results of the pair-wise comparison of probed names to differentiate a training data sample name pos_i a from negative sample name $neg_i \notin D_{train}$ based on the PER-tag prediction score computed by bert-base-NER and bert-large-NER. Each row represents the result for one input context, which is specified with the accuracy of the assumption that $\mu_{pos_i} > \mu_{neg_i}$ in each sample pair (pos_i, neg_i) and the number of True Positives and False Positives indicating which element of the respective sample pair achieved a higher PER-tag score. It is worth mentioning here, that due to the pair-wise approach the numbers for TP and TN, and FP and FN, are identical. This explains also

the equal values for precision and recall. The FP-rate can be thought of as a false alarm meter revealing how many neg_i pair elements are identified as training data samples due to their comparatively high prediction score.

Input Prompt c_i	Accuracy	TP	FP	Precision	Recall	FP-Rate
MASK	0.696	575	251	0.696	0.696	0.304
MASK is a person.	0.701	579	247	0.701	0.701	0.299
My name is MASK.	**0.723**	597	229	0.723	0.723	0.277
I am named MASK.	0.722	596	230	0.722	0.722	0.278
MASK is an individual human being.	0.702	580	246	0.702	0.702	0.298
Is MASK your name?	0.712	588	238	0.712	0.712	0.288
Is MASK a person?	0.685	566	260	0.685	0.685	0.315
MASK is not a person.	0.701	579	247	0.701	0.701	0.299
My name is not MASK.	0.706	583	243	0.706	0.706	0.294

Table 4.1.: Results for **bert-base-NER** with A_{dev} dataset

Input Prompt	Accuracy	TP	FP	Precision	Recall	FP-Rate
MASK	**0.707**	584	242	0.707	0.707	0.293
MASK is a person.	0.674	557	269	0.674	0.674	0.326
My name is MASK.	0.68	562	264	0.68	0.68	0.32
I am named MASK.	0.695	574	252	0.695	0.695	0.305
MASK is an individual human being.	0.674	557	269	0.674	0.674	0.326
Is MASK your name?	0.678	560	266	0.678	0.678	0.322
Is MASK a person?	0.684	565	261	0.684	0.684	0.316
MASK is not a person.	0.686	567	259	0.686	0.686	0.314
My name is not MASK.	0.668	552	274	0.668	0.668	0.332

Table 4.2.: Results for **bert-large-NER** with A_{dev} dataset

It can be seen from the accuracy ranging between 0.67 and 0.72 specified in Tables 4.1 and 4.2 that in a pair-wise comparison, a relatively high percentage of pos_i and neg_i across all input contexts are correctly classified. Interestingly, the accuracy varies quite considerably if we compare the results of the individual input contexts across model sizes. The best-performing input prompt using model size bert-base is “My name is MASK.” with an accuracy of 0.723, and for the model size bert-large, the best-performing prompt is “MASK” with an accuracy of 0.707, which is, however, the prompt with the highest FP-rate for bert-base-NER. For the larger NER model, the highest FP-rate can be observed with the input prompt “My name is not MASK.”. [Hisamoto et al., 2020] discovered for privacy attacks with a comparable approach that they naturally yield a high rate of false positives because of the overconfidence in neural networks. However, this claim may

be challenged by the possibility that potential overconfidence in predictions would also apply to training data samples and might be even more apparent in direct comparison with non-training data samples supporting the adopted assumption $\mu_{pos_i} > \mu_{neg_i}$. The pair-wise comparison of (pos_i, neg_i) inevitably results in a relatively high FP-rate, as a binary decision is made between two pair elements rather than over the entire sample.

4.1.1. Influence of Utterance Type and Syntactic Position of NEs

The type of utterance (assertion vs. interrogative vs. negation) and the syntactic position (subject vs. object) of the PER name in the input prompt only show minor effects in terms of accuracy. Figure 4.1 illustrates the influence of the type of utterance and syntactic position of the PER name in the prompt on the accuracy of the assumption $\mu_{pos_i} > \mu_{neg_i}$. A slight tendency to achieve a higher accuracy can be observed for the syntactic position of PER names in assertions, namely when the PER name is in object position (symbolized by a blue circle ●) in contrast to being in subject position (symbolized by a blue diamond ⧫). However, the number of observations per case[1] might be too small to draw meaningful conclusions from it.

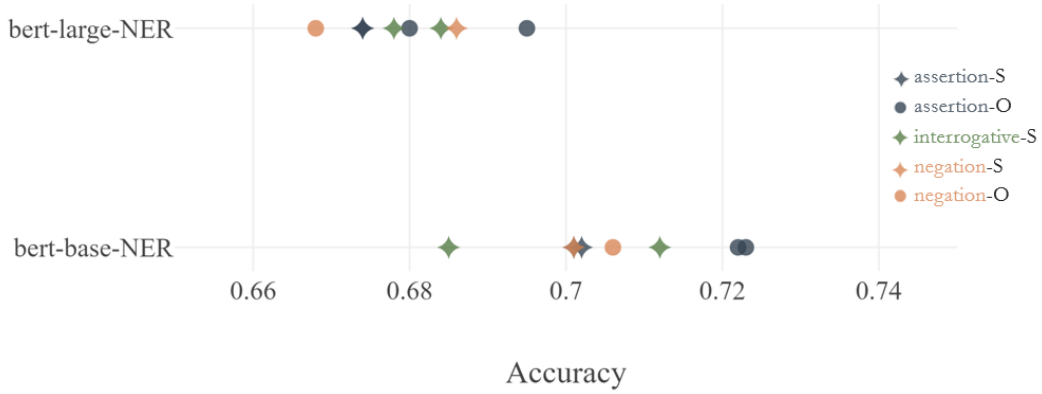

Figure 4.1.: Influence of type of utterance and syntactic position (subject/object) of PER name in the input prompt on the accuracy of $\mu_{pos_i} > \mu_{neg_i}$ per sample pair.

[1] Each combination of type of utterance + syntactic position of PER name occurs 1-2 times in the input prompt corpus C, as the total size of C is 9 and the input prompt c_1 "MASK" could not be considered in this comparison.

4.1.2. Scaling Phenomenon

The averaged results for both model sizes of bert-NER are presented in Table 4.3. Following [Yeom et al., 2018, Song and Raghunathan, 2020] this table includes a new metric. Adversarial advantage, sometimes also referred to as threat score, can be formalized as the difference between the true and false positive rate, i.e. TP-Rate $-$ FP-Rate. A higher adversarial advantage suggests that a model is more vulnerable to adversarial attacks, while random guessing provides 0 advantage [Song and Raghunathan, 2020]. With adversarial advantage scores above 0.36 for bert-NER, it can be argued that an adversary can gain a fair amount of information on the training data of bert-NER.

In summary, Table 4.3 reveals that the adopted method of taking the PER-tag prediction score as an indicator to distinguishing members and non-members of training data is more accurate for the smaller model bert-base-NER with an accuracy of 0.706 in comparison to bert-large-NER with an accuracy of 0.684. This finding refutes the "scaling phenomenon" proposed by [Carlini et al., 2022], which says that larger models memorize training data points to a greater extent.

Model Size	Accuracy	Recall	Precision	TP-Rate	FP-Rate	Adversarial Advantage
base	0.706	0.706	0.706	0.706	0.294	0.412
large	0.684	0.684	0.684	0.684	0.317	0.367

Table 4.3.: Comparison of $\mu_{pos_i} > \mu_{neg_i}$ accuracy for **bert-base-NER** and **bert-large-NER** over all input prompts with A_{dev} dataset.

4.1.3. Data Duplication

In addition to models scale, data duplication is also considered to be one of the key factors that significantly impact training data memorization by LLMs according to [Carlini et al., 2022]. It is argued that samples that occur multiple times in the training dataset are easier extractable in privacy attacks. From Figures 4.2 and 4.3 it can be observed for both model sizes of bert-NER that if a PER name occurs[2] at least six-times in the training data, the (averaged) prediction score for the PER-tag does not fall below a threshold of 0.85. However, the sample size of PER names with a high occurrence rate ($\geq$ six occurrences) in the training data is considerably lower than for the PER name samples with an occurrence rate under six. These results, therefore, need to be interpreted with caution. Overall, it

[2]As an occurrence counts the full multi-token name. One-token occurrences of just the first or last name are not included in the occurrence count.

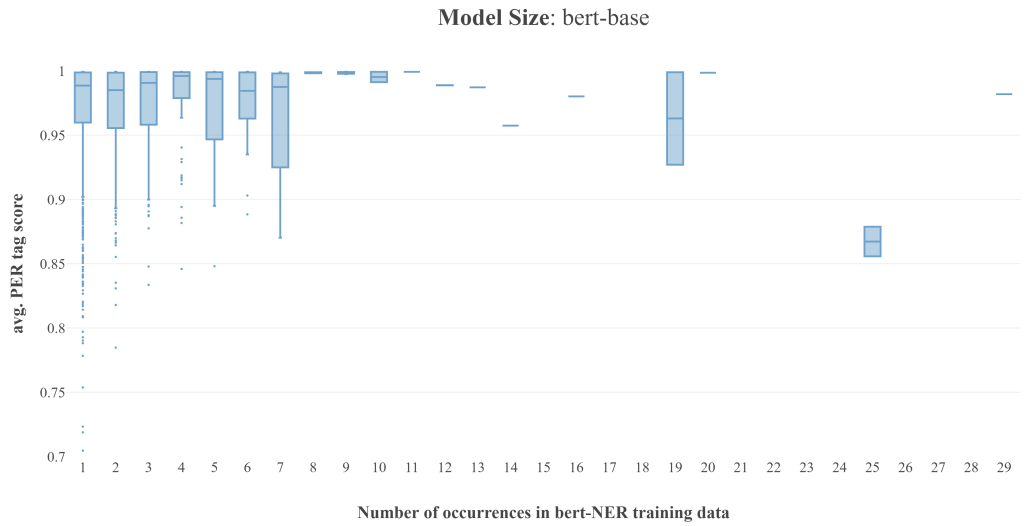

Figure 4.2.: Distribution of averaged bert-base-NER prediction output for PER names from the training dataset sorted by the PER name's occurrence count in the training data.

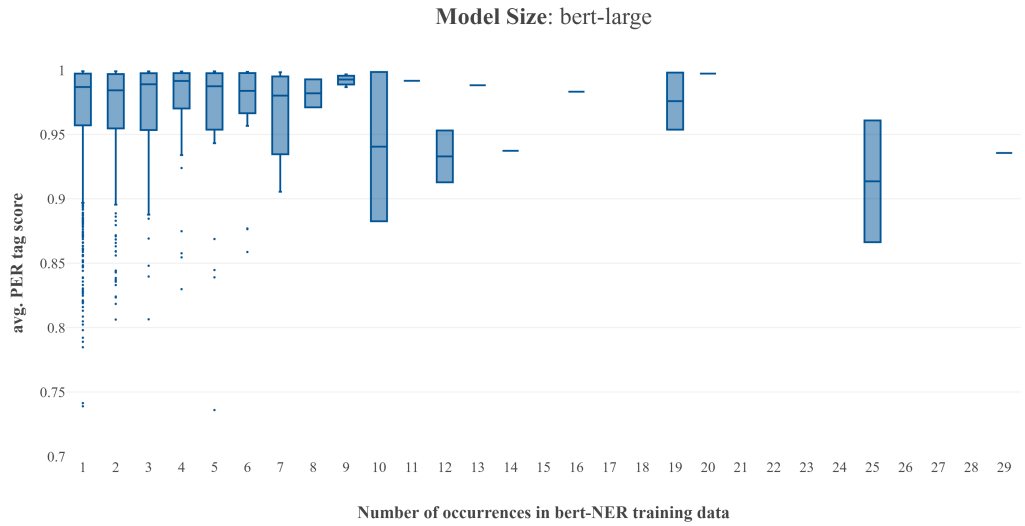

Figure 4.3.: Distribution of averaged bert-large-NER prediction output for PER names from the training dataset sorted by the PER name's occurrence count in the training data.

has to be noted that the box plots presented in Figures 4.2 and 4.3 vary in their number of observations, i.e., sole horizontal lines indicate that there was only one observation in this distribution. Similar to the comparison of accuracy and FP-rate among the two model sizes, it becomes evident that the smaller model bert-base-NER tends to assign higher PER-tag prediction scores than the larger model bert-large-NER at least for PER names occurring 15-times or less in the training data.

After having discussed the general insights into the prediction output of bert-NER, the following section will delve into the results of the simulated attack strategies for the (A) pair-wise and (B) full-list approach.

4.2. Results of Privacy Attack

This section presents the empirical analysis of the different attack strategies described in Section 3.3. For the simulated attack on the bert-NER model different approaches, (A) pair-wise comparison and (B) full-list collation, were tested to assess the risk of sensitive information leakage. The performance of the most effective strategy (A1-A3) and (B1-B2) for each approach was further verified on a test dataset.

4.2.1. Analysis of Pair-wise Comparison

Returning to the different attack strategies outlined in Section 3.3, the adversary Blake proposed three different methods to evaluate the data leakage potential of bert-NER for probes defined as pairs of PER names: **(A1)** Majority Voting, **(A2)** AVG Score and **(A3)** MAX Score. The adversary Charlie developed a random guessing approach as a baseline **(A4)** to validate Blake's strategies.

Figures 4.4 and 4.5 illustrate the count of sample pair wins per input context by the negative and positive samples of the paired probes in attack dataset A_{dev} for both model sizes. It can clearly be observed for all input contexts, that given a pair of sample names, the positive sample obtained a higher score in around two-thirds of the cases, which supports the hypothesis that samples seen during model training receive higher scores than unseen samples.

For the Majority Voting method, the PER-tag scores of the sample pairs (pos_i, neg_i) are compared per input prompt to decide on a winner name per context. The decisions are then pair-wise aggregated to ultimately determine the sample pair element, which is more likely to be a member of D_{train}. For the methods AVG Score and MAX Score, the PER-

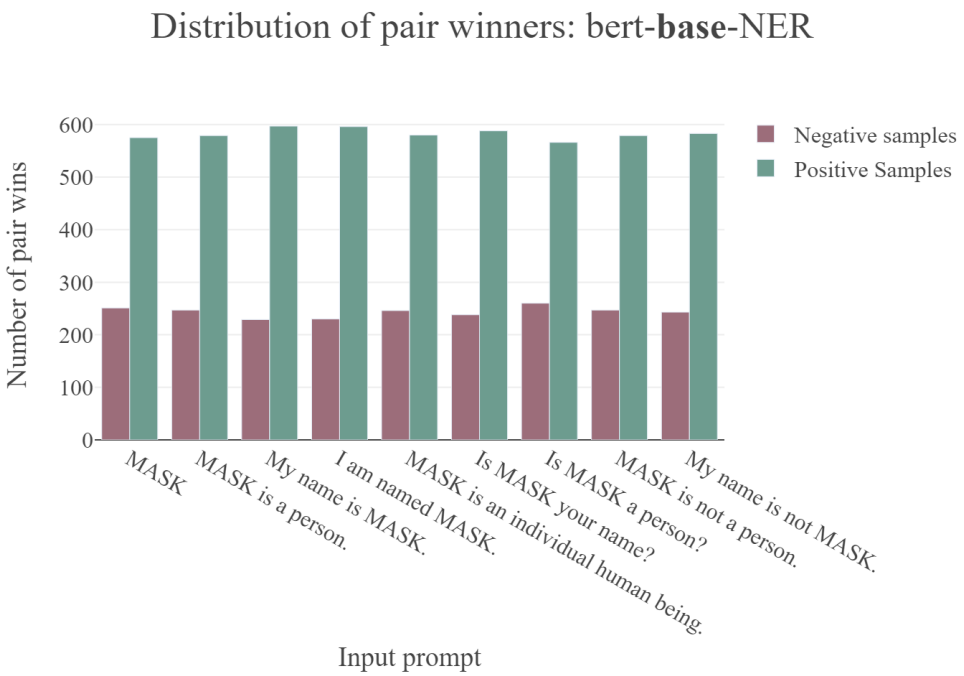

Figure 4.4.: Distribution of sample pair winners of attack dataset A_{dev} measured on the PER-tag prediction score produced by bert-base-NER.

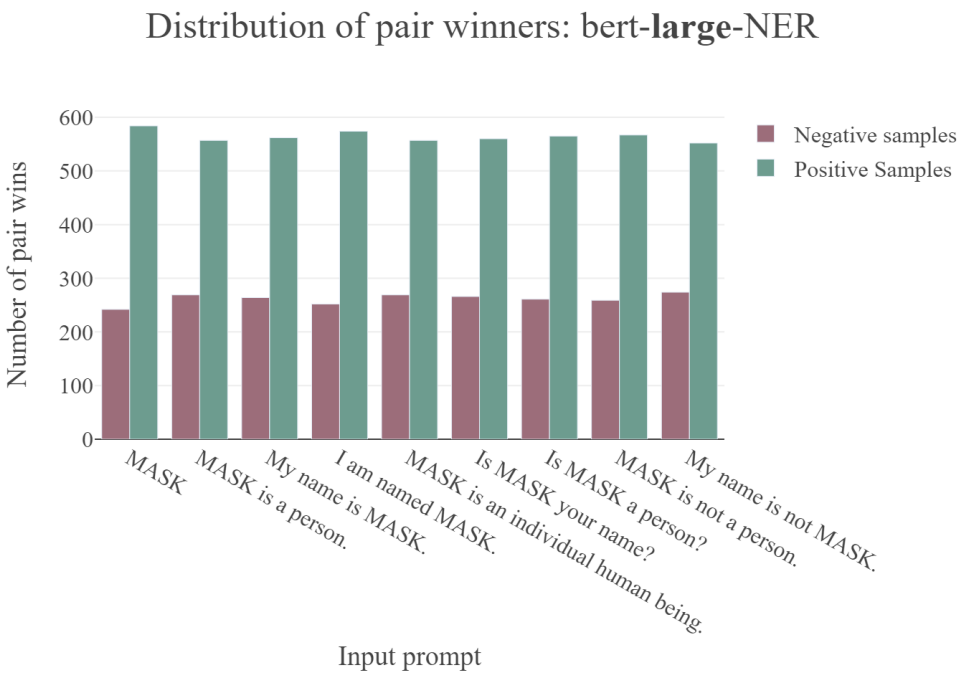

Figure 4.5.: Distribution of sample pair winners of attack dataset A_{dev} measured on the PER-tag prediction score produced by bert-large-NER.

tag scores for all input prompts per sample pair are adduced to ascertain the individual average score, or respectively, the maximum score, which consequently decides on the sample name expected to be a training data instance conforming to rules described in Section 3.3.

The results presented in Table 4.4 show that all attack strategies (A1)-(A3) exceed the threshold set by the baseline (A4), i.e. ≈50% accuracy. In [Hisamoto et al., 2020], an accuracy slightly above 50% in a binary classification task of this nature is already regarded as a potential breach of privacy. The margin to the baseline can be considered significant, as the respective accuracies lie more than 21 percentage points above the threshold for the bert-base-NER model and more than 17 percentage points above the threshold for the bert-large-NER model.

Method	**Accuracy:** bert-**base**-NER	**Accuracy:** bert-**large**-NER
(A1) Majority Voting	0.708	0.67
(A2) AVG Score	0.711	**0.686**
(A3) MAX Score	**0.721**	0.683
Baseline		Accuracy
(A4) Dummy Classifier		0.497

Table 4.4.: Comparison of the accuracy of methods (A1)-(A4) in assessing the leakage risk for bert-base-NER and bert-large-NER using the A_{dev} dataset.

Majority voting is the method with the lowest accuracy in identifying a training data instance among pairs of sample names for both model sizes. A possible explanation for the weaker performance of the majority voting strategy could be that the size of the input prompt corpus C might not have been sufficient enough to provide stronger predictions for the membership of PER names in D_{train}. The obtained results, however, can serve as a baseline for future investigations using an increased quantity of determinant context factors for majority voting. The other two methods investigated in the pair-wise approach to quantify the risk of information leakage demonstrate a model-dependent performance. For bert-large-NER, the accuracy of the AVG Score method is slightly higher than the MAX Score method. A greater margin in accuracy between these two methods, (A2) and (A3), can be found for the smaller bert-base-NER model. Here, the MAX Score method achieves an accuracy of 0.72, compared to an accuracy of 0.71 for the AVG Score method. As there is a moderately larger difference for the MAX Score strategy, this strategy will be further used for additional verification on test dataset A_{test}.

4.2.2. Analysis of Full-list Comparison

In addition to the pair-wise configuration of the attack dataset A, the dataset can also be used as a simple list of PER names. This list, composed of 50% positive samples pos_i and 50% negative samples neg_i, is exploited by the adversary Blake to assess sensitive information leakage from bert-NER based on two metrics: **(B1)** (Mean) Average Precision and **(B2)** Expected Rank of a randomly selected PER name sample from D_{train} given a ranked list of PER names. Similar to the pair-wise approach, a baseline **(B3)** to test the performance of the strategies was developed by the adversary Charlie.

Tables 4.5 and 4.6 list the rounded results for attack strategies (B1) and (B2) on bert-base-NER and bert-large-NER for each input prompt of corpus C. The average precision per input prompt is calculated by ranking the names according to their PER-tag-prediction score and iterating over this ordered list by applying the formula defined in 3.5. To compute the expected rank, i.e. the number of guesses until a randomly selected pos_i is encountered, the respective accuracies specified in Tables 4.1 and 4.2 are inserted into the formula given in 3.6. It should be noted that the expected rank values presented here are based on a ranking methodology that considers also other positive sample names occurring higher in the ranked list. For expected rank values that do not take into account the impact of other positive samples in the ranking, please refer to Tables A.1 and A.2 in the appendix.

As expected based on the knowledge acquired from the general finding section, the two model sizes show differences in the achieved results across the nine input prompts. For bert-base-NER the most revealing input context is “My name is MASK”, while for bert-large-NER “MASK”, the probed PER name without any context, can be identified as the most predictive in determining training data samples. Intuitively, the expected rank should be low when the AP is high and vice versa. This can be confirmed by the actual figures in Tables 4.5 and 4.6 and thus provides further evidence that the two metrics, expected rank and average precision, demonstrate informative value to evaluate data leakage by agreeing on the best performing input prompt.

Table 4.7 provides an overview of the results for the attack strategies (B1), (B2) and the baseline (B3) for the full-list approach. The average precision attempts to quantify the attack’s total performance, taking into consideration both the strategy’s capacity to detect relevant instances and the quality of those detections. Here, this means that the aim is to find a high number of pos_i samples. The Mean Average Precision for bert-base-NER and bert-large-NER is calculated as the arithmetic mean of the AP scores for the individual input prompts. An alternative approach to taking the mean of the AP scores defined in

Input Prompt c_i	**(B1) Avg. Precision**	**(B2) Expected Rank**
MASK	0.697	664
MASK is a person.	0.729	661
My name is MASK.	**0.731**	**642**
I am named MASK.	0.721	643
MASK is an individual human being.	0.721	660
Is MASK your name?	0.715	652
Is MASK a person?	0.699	674
MASK is not a person.	0.726	661
My name is not MASK.	0.724	656

Table 4.5.: Attack evaluation metrics for **bert-base-NER** with A_{dev} dataset

Input Prompt c_i	**(B1) Avg. Precision**	**(B2) Expected Rank**
MASK	**0.707**	**655**
MASK is a person.	0.7	677
My name is MASK.	0.695	677
I am named MASK.	0.702	666
MASK is an individual human being.	0.693	683
Is MASK your name?	0.699	679
Is MASK a person?	0.703	675
MASK is not a person.	0.7	673
My name is not MASK.	0.69	687

Table 4.6.: Attack evaluation metrics for **bert-large-NER** with A_{dev} dataset

Tables 4.5 and 4.6 could be to calculate the average precision from the ordered list of averaged PER-tag scores over all nine contexts, i.e. total mean PER-tag score for each sample name. The comparison of these two values, MAP and AP, in Table 4.7 shows that the second approach, AP, yields marginally better results, especially for the larger model bert-large-NER improving the score by 1%.

Model Size	**(B1) MAP**	**(B1) AP**	**(B2) Expected Rank**
base	0.718	0.723	656
large	0.699	0.708	675
Baseline			**(B3) Accuracy**
Dummy Classifier			0.497

Table 4.7.: Comparison of bert-base-NER and bert-large-NER leakage assessment metrics using the A_{dev} dataset with averaged scores over all input prompts c_i for Average Precision and Expected Rank and for MAP the arithmetic mean of the AP scores from Tables 4.5 and 4.6.

The attack strategy (B2) approaches the issue of estimating data leakage in a LLM from a different perspective. The goal for the expected rank metric is to achieve a value as low as possible in order to define the attack as successful. At first sight, the numbers in Tables 4.5, 4.6 and 4.7 may appear relatively high in comparison to the sample size (= 1653), but the results need to be interpreted under the condition that the formula given in 3.6 approximates the rank of a specific randomly selected sample from $D\prime$ and not of just any, or first, positive sample in a ranked list. Put differently, for a particular sample pos_i to be encountered in an ordered list of PER names, the expected value of name guesses required is 656 (bert-base-NER), or alternatively 675 (bert-large-NER). Due to the lack of comparable approaches, or other referential work, it is challenging to set a benchmark to distinguish between satisfactory and unsatisfactory scores for the expected rank metric. As previously with the individual input prompt dependent results, the expected rank values without considering other positive samples in the ranking methodology can be found in Table A.3 in the appendix.

In comparison to attack strategy (B2) for the full-list approach, using average precision offers a number of advantages. First, average precision is a well-established and frequently used measure in a number of different areas of application. It can be easily compared to a defined baseline and thus offers a higher level of reliability compared to the expected rank method. An additional benefit of using average precision is that it can also be applied to smaller datasets. Based on these pragmatic considerations, the attack strategy (B1) involving average precision was chosen for the verification process with dataset A_{test}.

4.2.3. Verification on Test Dataset

This section aims to verify the effectiveness of one promising strategy per approach (A and B) out of the multiple strategies defined in the leakage risk analysis framework for bert-NER. For the pair-wise approach (A), the focus lies on the MAX Score strategy and for the full-list approach (B), Average Precision is used to asses the extent of data leakage risk.

(A) Pair-wise Comparison: MAX Score

Figure 4.6 attempts to display the individual MAX scores per sample name pair, which are used to determine the training data instance within each pair according to the formula defined under **(A3)**. The purpose of this figure is to observe general trends, so that the values of singular samples can be neglected. Naturally, the PER-Tag MAX scores for the PER names in the attack dataset are located more towards the upper end of the scale,

i.e. most of the MAX scores accumulate in the area above 0.9 in both model sizes. When comparing the MAX scores between model sizes it becomes evident that number of MAX scores below a PER-tag prediction score of 0.85 is moderately higher in case of the larger model bert-large-NER, which supports the overall finding that bert-large-NER assigns lower prediction scores in general and is therefore less susceptible to privacy attacks based on prediction scores than bert-base-NER.

Figure 4.6.: Distribution of MAX scores for pos_i and neg_i per sample name pair as formulated under **(A3)** using dataset A_{test}.

Table 4.8 displays the accuracy of the MAX Score strategy in determining training data samples given a pair of sample names (pos_i, neg_i) for both model sizes of bert-NER (cf. Table 4.4 for the performance of the MAX Score strategy on the A_{dev} dataset). It can be seen that the accuracy of the MAX score strategy is considerably higher than the baseline set by the dummy classifier (A4). In comparison to the accuracies achieved using the A_{dev} dataset the performance, however, shows a minor drop for both model sizes alike. Overall, the values in Table 4.8 suggest a similar observation as was found for the A_{dev} dataset, the smaller model bert-base-NER exhibits a greater vulnerability to training data leakage than the larger model bert-large-NER.

Method	**Accuracy:** bert-**base**-NER	**Accuracy:** bert-**large**-NER
(A3) MAX Score	0.712	0.656
Baseline		Accuracy
(A4) Dummy Classifier		0.497

Table 4.8.: Verification of attack strategy (A3) MAX Score using A_{test} dataset.

(B) Full-List Comparison: Average Precision

Figure 4.7 demonstrates the average precision achieved for each context input for both bert-NER sizes (see Table A.4 in the appendix for the numerical data). Despite the lower AP scores for the A_{test} dataset compared to the A_{dev} dataset, the performance of the attack strategy is still encouraging and suggest that it can be successfully deployed in a data leakage risk assessment. Similar to the observation for the A_{dev} dataset, the average precision for the smaller model, bert-base-NER, seems to be generally higher than for the larger model bert-large-NER. Except for the input context c_7 "Is MASK a person?", the average precision scores of the privacy attack for the two model sizes are separated by a difference of at least 1.5 percentage points. Contexts c_1 to c_6 show a similar trend in both model sizes regarding the magnitude of AP. Contexts c_8 and c_9, on the other hand, have an entirely different impact on the attack evaluation metric. In case of bert-base-NER, the two input contexts of utterance type negation increase the overall AP, whereas in case of bert-large-NER they have the opposite effect. Given that this finding is based on a single NER model in two sizes, further experimental investigations are needed to examine whether the utterance type of input prompts influences the output of models with different sizes.

Model Size	**MAP**	**AP**
base	0.689	0.701
large	0.664	0.671
Baseline	**(B3) Accuracy**	
Dummy Classifier	0.498	

Table 4.9.: Comparison of bert-base-NER and bert-large-NER on the attack evaluation metric Mean Average Precision/Average Precision using the verification dataset A_{test}.

The aggregated verification results for bert-NER are shown in Table 4.9. In line with the per-context results, bert-base-NER appears to have a higher risk of leaking sensitive information in the form of PER names as bert-large-NER. A margin of 0.025 for the Mean Average Precision and even larger margin of 0.03 for the average precision considering

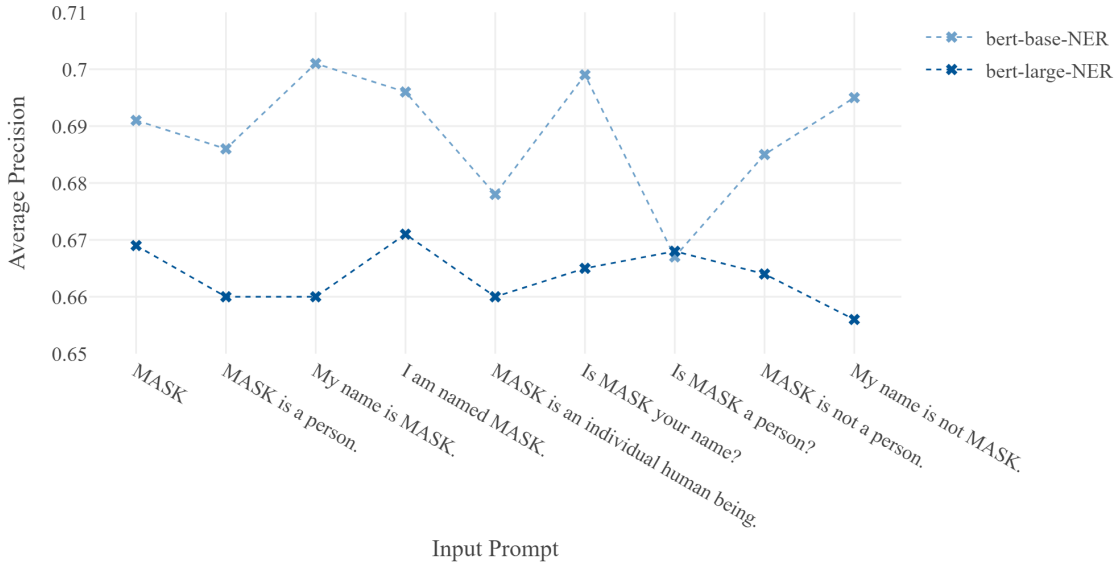

Figure 4.7.: Verification of attack strategy B1 using dataset A_{test}

averaged PER-tag scores can be observed between the metrics documented for the two model sizes. Consistent with the results from A_{dev}, the AP computed from a list ranked with mean PER-tag scores achieves a higher score than the simple arithmetic mean across individual AP scores, i.e. MAP. Overall, the outcome of the verification with A_{test} is slightly less strong with differences to the MAP and AP scores using A_{dev} ranging from 0.22 to 0.37. Despite the minor decrease in absolute values with the test dataset, MAP and AP in connection with the prediction output for PER-tag scores proves to be a robust strategy to estimate sensitive information leakage in bert-NER.

The findings presented in this chapter reinforce the assumption that training data samples tendentially receive higher output scores than samples previously unseen by the model. Further, the effectiveness of the strategies (A3) MAX Score and (B1) Average Precision as a measure to assess the risk of data leakage in LLMs could be validated. However, the effect of model size in this context requires further attention since the results for bert-NER turned out differently than what was anticipated from the literature. Taken together, the findings for all attack strategies tested in the analytical risk framework proposed in this thesis do not support previous research claiming that larger models are more prone to training data leakage [Carlini et al., 2022]. In fact, for bert-NER the opposite appears to hold true. The underlying causes for this phenomenon in bert-NER are yet to be determined in future work.

5. Limitations and Outlook

This section of the thesis provides an outlook on the implications of the findings presented in previous sections. It includes a discussion on the limitations of the study and potential for further research. The practical significance of the results for the field of data leakage and privacy protection are also highlighted. Furthermore, it provides a summary of the main contributions of the thesis, their significance in the current state of knowledge, and briefly describes approaches attempting to mitigate the identified privacy risks in language models.

Limitations

While model output or score-based privacy attacks may be simple and easy to implement and therefore constitute an adversary-friendly approach to obtain information on the training data of Large Language Models, the issue of overconfident predictions of neural networks needs to be addressed. This overconfidence can pose a challenge as it may lead to distorted results and potentially generate an increased number of false positives, i.e. data samples falsely detected as members of the model's training dataset [Hintersdorf et al., 2021].

This research is further limited by the lack of information on the scale of acceptability for data leakages in LLMs. Due to the absence of a formally defined reference framework or benchmark specifying the extent and nature of data leakage considered considered acceptable or unacceptable, it is difficult to classify the performance of the targeted models outside of the self-imposed guidelines established within this study. In a highly restrictive interpretation, any kind of leakage would be considered unacceptable. This, however, appears to be an unrealistic approach and would presumably prevent the deployment of any current LLM.

Future Work

This research has raised several questions in need of further investigation. For instance, future work could explore the impact of model scale on information leakage in LLMs, as the results on this topic in this thesis diverge from previous findings (cf. [Carlini et al., 2022]). This difference may be attributed to the different types of LLMs that were examined in this regard, neural tagger vs. generative LLM, or there may be other underlying reasons to this scaling phenomenon. Another interesting research direction would involve testing the effect of different types of input prompts on an output-based privacy attack in more detail and also analyzing the influence of an increased number of input prompts on the proposed attack strategies. Through the use of a generative AI system the creation of input prompts could be automated and thereby extending or even replacing the manually produced corpus. Additionally, future studies could concentrate on information leakage in pre-trained models and their fine-tuned versions to investigate whether there is a significant difference concerning vulnerability to privacy attacks in the two steps of model development.

In this work, the focus was on detecting leakages from LLMs in the form of sensitive information, such as Personally Identifiable Information. While identifying the vulnerability of AI systems to privacy breaches is essential to estimate the risk in deployment, it is equally important to consider protective measures earlier in the development process. On a higher level, privacy-preserving techniques such as differential privacy [Dwork et al., 2017] and federated learning [McMahan et al., 2016] have been proposed to protect the privacy of sensitive data points in a dataset, when data is shared or used for data analysis or machine learning applications. Lower level, "data-near" techniques, such as data aggregation [He et al., 2011], encryption [Gai et al., 2021] or obfuscation [Brunton, 2015] seek to preserve data privacy by rendering it unreadable or incomprehensible to unauthorized parties. Although, there already exists a toolbox to protect data privacy, there are still open questions and challenges. More research is especially needed on how to implement the proposed privacy-enhancing measures in the context of large amounts of web-scraped data, as current solutions may be impractical in this regard [Wallace et al., 2020]. Overall, privacy protection techniques present numerous opportunities for further investigations, but they fall outside the scope of this thesis which concentrated on detecting privacy leakage.

6. Conclusion

Identifying the risk of data leakage in Large Language Models (LLMs) has become increasingly important considering both, the growing use of LLMs for personal and professional purposes, and the potential consequences of unauthorized or unintentional exposure of (sensitive) information. LLMs require vast amounts of training data from different domains to produce high-quality output. This data can be sourced from publicly accessible data from the internet or private datasets, which both may include sensitive information, such as Personally Identifiable Information (PII). Therefore, the leakage of sensitive data from LLMs poses a realistic threat and has been recognized as such in the research field of artificial intelligence. This study aimed to explore sensitive information leakage in neural taggers, a subcategory of LLMs, and endeavoured to contribute to the existing body of knowledge on the subject.

This thesis has addressed the question whether the prediction score of a neural tagger represents a dependable indicator to differentiate between training data and non-training data samples. Furthermore, the risk of sensitive information leakage, in specific Person (PER) names, in a BERT-based Named Entity Recognition (NER) model was estimated by developing a multi-approach risk analysis framework and applying different attack strategies based on the PER-tag prediction output provided by the model. The underlying assumption here was that training data instances receive higher prediction scores than data samples which were not part of the training dataset and thus, have not been encountered by the model during the training phase.

In summary, the findings of this study support the assumption that training data instances tend to receive higher tag prediction scores than non-training data samples. This was shown by a pair-wise comparison of PER names from the training dataset and out-of-sample PER names not used during model training, in which approximately 70 % of sample pairs included a training data sample with a higher PER-tag prediction. In this context, the phenomenon of training data memorization in connection with model scale and data duplication in the training dataset was also investigated. In contrast to previous research work, no clear correlation between frequency of occurrence in the training dataset and

PER-tag prediction could be established in this study. For the influence of the model scale on training data memorization, in terms of a high(er) PER-tag prediction score, it was found that the smaller model size of the targeted NER model leaks more data than the larger one.

The analytical risk assessment framework intends to simulate a black-box, passive, supervised privacy attack on the target NER model employing various strategies. By using a broad range of strategies, it was possible to investigate different methods and metrics for estimating the vulnerability to training data leakage in a structured process. The results of the experiments indicate that it is generally possible to distinguish between training and non-training PER names based on the model's PER-tag prediction score. All of the attack strategies employed in the leakage risk framework performed better than the baseline methods. However, it should be noted here that not all strategies are metrically comparable to the defined baseline. For this reason and other practical considerations only two of the five attack strategies defined in the framework were further used to verify the results on a second attack dataset. The results obtained from the test dataset confirmed that the attack strategies (A3) MAX Score and (B1) Average Precision are effective in evaluating sensitive information leakage in the targeted model. In addition, it was shown that different model sizes or models can be compared with each other using relatively simple means and without requiring access to the models' (hyper-)parameters. Taken together, the findings presented in this thesis add to a rapidly expanding field of leakage risk assessment for LLMs aimed at evaluating the extent of (sensitive) information leakage.

Ethics Statement

During the research for this thesis, only publicly available data on the internet was used. In the case of the CoNLL 2003 dataset, additional measures were taken to ensure ethical use of the data. These measures included obtaining explicit consent from the Reuters news agency to use the dataset in this research, and ensuring that all results and analyses were reported in an aggregated or anonymous manner to protect the privacy of any individuals and organizations that may be identified in the data. In cases where specific names were used for demonstrative purposes, they were not names of private individuals but names of fictional characters.

Bibliography

[Adamopoulou and Moussiades, 2020] Adamopoulou, E. and Moussiades, L. (2020). Chatbots: History, technology, and applications. *Machine Learning with Applications*, 2:100006.

[Anderson, 2008] Anderson, R. (2008). *Security engineering : a guide to building dependable distributed systems*. Wiley, Indianapolis, Ind. [u.a.], 2. ed.. edition.

[Balle et al., 2022] Balle, B., Cherubin, G., and Hayes, J. (2022). Reconstructing training data with informed adversaries. *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1138–1156.

[Bender and Friedman, 2018] Bender, E. M. and Friedman, B. (2018). Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

[Bender et al., 2021] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

[Bengio et al., 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

[Bhilare et al., 2009] Bhilare, D., Ramani, A., and Tanwani, S. (2009). Protecting intellectual property and sensitive information in academic campuses from trusted insiders: leveraging active directory. In *Proceedings of the 37th annual ACM SIGUCCS fall conference*, SIGUCCS '09, pages 99–104. ACM.

[Boddington, 2017] Boddington, P. (2017). *Towards a code of ethics for artificial intelligence*. Artificial intelligence: foundations, theory, and algorithms. Springer, Cham.

[Bottou et al., 2016] Bottou, L., Curtis, F. E., and Nocedal, J. (2016). Optimization methods for large-scale machine learning. *SIAM Rev.*, 60:223–311.

[Brown et al., 2022] Brown, H., Lee, K., Mireshghallah, F., Shokri, R., and Tramèr, F. (2022). What does it mean for a language model to preserve privacy? In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 2280–2292, New York, NY, USA. Association for Computing Machinery.

[Brown and Miller, 2013] Brown, K. and Miller, J. (2013). Proper noun. In *The Cambridge Dictionary of Linguistics*, page 3–2. Cambridge University Press.

[Brown et al., 1993] Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics - Association for Computational Linguistics*, 19(2):263–311.

[Brown et al., 2020] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan,

T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners.

[Brundage et al., 2018] Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., Dafoe, A., Scharre, P., Zeitzoff, T., Filar, B., Anderson, H., Roff, H., Allen, G. C., Steinhardt, J., Flynn, C., hÉigeartaigh, S. , Beard, S., Belfield, H., Farquhar, S., Lyle, C., Crootof, R., Evans, O., Page, M., Bryson, J., Yampolskiy, R., and Amodei, D. (2018). The malicious use of artificial intelligence: Forecasting, prevention, and mitigation.

[Brunton, 2015] Brunton, F. (2015). *Obfuscation : a user's guide for privacy and protest /*. The MIT Press„ Cambridge, Massachusetts ; London, England :.

[Carlini et al., 2022] Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramer, F., and Zhang, C. (2022). Quantifying memorization across neural language models.

[Carlini et al., 2018] Carlini, N., Liu, C., Erlingsson, , Kos, J., and Song, D. (2018). The secret sharer: Evaluating and testing unintended memorization in neural networks.

[Carlini et al., 2020] Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., Oprea, A., and Raffel, C. (2020). Extracting training data from large language models.

[Chowdhery et al., 2022] Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. (2022). Palm: Scaling language modeling with pathways.

[Clark et al., 2013] Clark, A., Fox, C., and Lappin, S. (2013). *The handbook of computational linguistics and natural language processing*. Blackwell Handbooks in Linguistics. Wiley-Blackwell, West Sussex, England.

[Code page, 1252] Code page (1252). Window's code pages. https://learn.microsoft.com/en-us/cpp/c-runtime-library/code-pages?view=msvc-170. Online, accessed July, 31st.2022.

[Collins Dictionary,] Collins Dictionary. person. https://www.collinsdictionary.com/dictionary/english/person. Online, accessed July, 1st.2022.

[Croft et al., 2009] Croft, W. B., Metzler, D., and Strohman, T. (2009). Search engines - information retrieval in practice.

[Dabre et al., 2020] Dabre, R., Chu, C., and Kunchukuttan, A. (2020). A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5).

[Dargan et al., 2020] Dargan, S., Kumar, M., Ayyagari, M. R., and Kumar, G. (2020). A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, 27(4):1071–1092.

[Deng and Liu, 2018] Deng, L. and Liu, Y. (2018). *Deep learning in natural language processing*. Springer.

[Derczynski et al., 2016] Derczynski, L., Bontcheva, K., and Roberts, I. (2016). Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan. The COLING 2016 Organizing Committee.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

[Dietterich, 2000] Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*.

[Dwork et al., 2017] Dwork, C., Smith, A., Steinke, T., and Ullman, J. (2017). Exposed! a survey of attacks on private data. *Annual review of statistics and its application*, 4(1):61–84.

[Erhan et al., 2010] Erhan, D., Courville, A., Bengio, Y., and Vincent, P. (2010). Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings.

[European Commission of Justice and Consumers, 2016] European Commission of Justice and Consumers (2016). Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation). https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679#d1e1374-1-1. Online, accessed January, 6^{th}.2023.

[Färber et al., 2017] Färber, M., Bartscherer, F., Menne, C., and Rettinger, A. (2017). Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129.

[Gai et al., 2021] Gai, K., Qiu, M., and Zhao, H. (2021). Privacy-preserving data encryption strategy for big data in mobile cloud computing. *IEEE transactions on big data*, 7(4):678–688.

[Gebru et al., 2021] Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., III, H. D., and Crawford, K. (2021). Datasheets for datasets. *Commun. ACM*, 64(12):86–92.

[Gehman et al., 2020] Gehman, S., Gururangan, S., Sap, M., Choi, Y., and Smith, N. A. (2020). RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Softmax Units for Multinoulli Output Distributions*, chapter 6.2.2.3, pages 180–184. The MIT Press. http://www.deeplearningbook.org.

[Goutte and Gaussier, 2005] Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In Losada, D. E. and Fernández-Luna, J. M., editors, *Advances in Information Retrieval*, pages 345–359, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Government of Brazil, 2018] Government of Brazil (2018). Lei geral de proteção de dados. https://www.gov.br/inss/pt-br/acesso-a-informacao/lei-geral-de-protecao-de-dados-pessoais. Online, accessed January, 10^{th}.2023.

[He et al., 2011] He, W., Liu, X., Nguyen, H. V., Nahrstedt, K., and Abdelzaher, T. (2011). Pda: Privacy-preserving data aggregation for information collection. *ACM Trans. Sen. Netw.*, 8(1).

[Hiemstra, 2009] Hiemstra, D. (2009). *Language Models*, pages 1591–1594. Springer US, Boston, MA.

[Hintersdorf et al., 2021] Hintersdorf, D., Struppek, L., and Kersting, K. (2021). To trust or not to trust prediction scores for membership inference attacks.

[Hisamoto et al., 2020] Hisamoto, S., Post, M., and Duh, K. (2020). Membership inference attacks on sequence-to-sequence models: Is my data in your machine translation system? *Transactions of the Association for Computational Linguistics*, 8:49–63.

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

[Jobin et al., 2019] Jobin, A., Ienca, M., and Vayena, E. (2019). Artificial intelligence: the global landscape of ethics guidelines. *arXiv.org*.

[Jurafsky and Martin, 2009] Jurafsky, D. and Martin, J. H. (2009). *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Education, Upper Saddle River, NJ, 2. ed., pearson internat. ed. edition.

[Jurafsky and Martin, 2023] Jurafsky, D. and Martin, J. H. (2023). Speech and language processing. https://web.stanford.edu/~jurafsky/slp3/ed3book_jan72023.pdf. Draft version, accessed online February, 12^{th}.2023.

[Kreutzer et al., 2022] Kreutzer, J., Caswell, I., Wang, L., Wahab, A., van Esch, D., Ulzii-Orshikh, N., Tapo, A., Subramani, N., Sokolov, A., Sikasote, C., Setyawan, M., Sarin, S., Samb, S., Sagot, B., Rivera, C., Rios, A., Papadimitriou, I., Osei, S., Suarez, P. O., Orife, I., Ogueji, K., Rubungo, A. N., Nguyen, T. Q., Müller, M., Müller, A., Muhammad, S. H., Muhammad, N., Mnyakeni, A., Mirzakhalov, J., Matangira, T., Leong, C., Lawson, N., Kudugunta, S., Jernite, Y., Jenny, M., Firat, O., Dossou, B. F. P., Dlamini, S., de Silva, N., Çabuk Ballı, S., Biderman, S., Battisti, A., Baruwa, A., Bapna, A., Baljekar, P., Azime, I. A., Awokoya, A., Ataman, D., Ahia, O., Ahia, O., Agrawal, S., and Adeyemi, M. (2022). Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

[Lam and Suen, 1997] Lam, L. and Suen, S. (1997). Application of majority voting to pattern recognition: an analysis of its behavior and performance. *IEEE transactions on systems, man and cybernetics. Part A, Systems and humans*, 27(5):553–568.

[Lehman et al., 2021] Lehman, E., Jain, S., Pichotta, K., Goldberg, Y., and Wallace, B. C. (2021). Does BERT pretrained on clinical notes reveal sensitive data? *CoRR*, abs/2104.07762.

[Lever et al., 2016] Lever, J., Krzywinski, M., and Altman, N. (2016). Points of significance: Model selection and overfitting. *Nature methods*, 13(9):703.

[Lewis et al., 2004] Lewis, D. D., Yang, Y., Russell-Rose, T., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

[Li et al., 2022] Li, J., Sun, A., Han, J., and Li, C. (2022). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

[Li et al., 2020] Li, K., Liu, Z., He, T., Huang, H., Peng, F., Povey, D., and Khudanpur, S. (2020). An empirical study of transformer-based neural language model adaptation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7934–7938.

[Lim, 2020a] Lim, D. S. (2020a). bert-base-ner. https://huggingface.co/dslim/bert-base-NER. Online, retrieved July, 6^{th}.2022.

[Lim, 2020b] Lim, D. S. (2020b). bert-large-ner. https://huggingface.co/dslim/bert-large-NER. Online, retrieved July, 6^{th}.2022.

[Lison et al., 2021] Lison, P., Pilán, I., Sanchez, D., Batet, M., and Øvrelid, L. (2021). Anonymisation models for text data: State of the art, challenges and future directions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4188–4203, Online. Association for Computational Linguistics.

[Liu et al., 2021] Liu, B., Ding, M., Shaham, S., Rahayu, W., Farokhi, F., and Lin, Z. (2021). When machine learning meets privacy: A survey and outlook. *ACM Comput. Surv.*, 54(2).

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

[McCallum et al., 1998] McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Madison, WI.

[McMahan et al., 2017] McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In Singh, A. and Zhu, J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

[McMahan et al., 2016] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. (2016). Communication-efficient learning of deep networks from decentralized data.

[Min et al., 2021] Min, B., Ross, H., Sulem, E., Veyseh, A. P. B., Nguyen, T. H., Sainz, O., Agirre, E., Heintz, I., and Roth, D. (2021). Recent advances in natural language processing via large pre-trained language models: A survey. *CoRR*, abs/2111.01243.

[Mireshghallah et al., 2020] Mireshghallah, F., Taram, M., Vepakomma, P., Singh, A., Raskar, R., and Esmaeilzadeh, H. (2020). Privacy in deep learning: A survey. *CoRR*, abs/2004.12254.

[Murakonda and Shokri, 2020] Murakonda, S. K. and Shokri, R. (2020). Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning.

[Nakamura et al., 2021] Nakamura, Y., Hanaoka, S., Nomura, Y., Hayashi, N., Abe, O., Yada, S., Wakamiya, S., and Aramaki, E. (2021). KART: privacy leakage framework of language models pre-trained with clinical records. *CoRR*, abs/2101.00036.

[Nasr et al., 2019] Nasr, M., Shokri, R., and Houmansadr, A. (2019). Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, volume 2019-, pages 739–753, Ithaca. IEEE.

[Oh et al., 2017] Oh, S. J., Augustin, M., Fritz, M., and Schiele, B. (2017). Towards reverse-engineering black-box neural networks. In *International Conference on Learning Representations*.

[OpenAI, 2023] OpenAI (2023). Gpt-4 technical report.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv.org*.

[PreTrainedTokenizerFast, version 4.26.0,] PreTrainedTokenizerFast, version 4.26.0. https://huggingface.co/docs/transformers/v4.26.0/en/main_classes/tokenizer#transformers.PreTrainedTokenizerFast. Online, accessed January, 4^{th}.2023.

[Qiu et al., 2020] Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., and Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China. Technological sciences*, 63(10):1872–1897.

[Quinn and Malgieri, 2021] Quinn, P. and Malgieri, G. (2021). The difficulty of defining sensitive data—the concept of sensitive data in the eu data protection framework. *German Law Journal*, 22(8):1583–1612.

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

[Ramshaw and Marcus, 1995] Ramshaw, L. and Marcus, M. (1995). Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

[Raschka, 2023] Raschka, S. (2023). Understanding and coding the self-attention mechanism of large language models from scratch. https://sebastianraschka.com/blog/2023/self-attention-from-scratch.html. Online, accessed February, 10^{th}.2023.

[Rei et al., 2016] Rei, M., Crichton, G. K. O., and Pyysalo, S. (2016). Attending to characters in neural sequence labeling models. *CoRR*, abs/1611.04361.

[Rigaki and Garcia, 2020] Rigaki, M. and Garcia, S. (2020). A survey of privacy attacks in machine learning. *CoRR*, abs/2007.07646.

[Salem et al., 2019] Salem, A., Bhattacharyya, A., Backes, M., Fritz, M., and Zhang, Y. (2019). Updates-leak: Data set inference and reconstruction attacks in online learning. *ArXiv*, abs/1904.01067.

[Samek et al., 2017] Samek, W., Wiegand, T., and Müller, K. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *CoRR*, abs/1708.08296.

[Sang and Meulder, 2003] Sang, E. F. T. K. and Meulder, F. D. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

[Schmidhuber, 2015] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.

[Scholz, 2019] Scholz, P., editor (2019). *Information security: cyberattacks, data breaches and security controls*. Privacy and identity protection. Nova Science Publishers, Incorporated, New York.

[Sennrich et al., 2015] Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units.

[Sharma et al., 2020] Sharma, S., Sharma, S., and Athaiya, A. (2020). Activation functions in neural networks. *International Journal of Engineering Applied Sciences and Technology*, 4(12):310–316.

[Shokri et al., 2017] Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

[Smith et al., 2022] Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., Zhang, E., Child, R., Aminabadi, R. Y., Bernauer, J., Song, X., Shoeybi, M., He, Y., Houston, M., Tiwary, S., and Catanzaro, B. (2022). Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model.

[Sokolova and Lapalme, 2009] Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing Management*, 45(4):427–437.

[Song and Raghunathan, 2020] Song, C. and Raghunathan, A. (2020). Information leakage in embedding models. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 377–390.

[Song et al., 2017] Song, C., Ristenpart, T., and Shmatikov, V. (2017). Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 587–601, New York, NY, USA. Association for Computing Machinery.

[State of California, 2018] State of California (2018). California consumer privacy act. https://www.oag.ca.gov/privacy/ccpa/. Online, accessed January, 10^{th}.2023.

[Tamkin et al., 2021] Tamkin, A., Brundage, M., Clark, J., and Ganguli, D. (2021). Understanding the capabilities, limitations, and societal impact of large language models.

[Thomas et al., 2020] Thomas, A., Adelani, D. I., Davody, A., Mogadala, A., and Klakow, D. (2020). Investigating the impact of pre-trained word embeddings on memorization in neural networks. In *Text, Speech, and Dialogue*, Lecture Notes in Computer Science, pages 273–281, Cham. Springer International Publishing.

[Tirumala et al., 2022] Tirumala, K., Markosyan, A. H., Zettlemoyer, L., and Aghajanyan, A. (2022). Memorization without overfitting: Analyzing the training dynamics of large language models.

[Tramèr et al., 2016] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. (2016). Stealing machine learning models via prediction apis. *ArXiv*, abs/1609.02943.

[Vakili and Dalianis, 2021] Vakili, T. and Dalianis, H. (2021). Are clinical bert models privacy preserving? the difficulty of extracting patient-condition associations. In *Proceedings of the AAAI 2021 Fall Symposium on Human Partnership with Medical AI*, CEUR Workshop Proceedings.

[van der Waa et al., 2020] van der Waa, J., Schoonderwoerd, T., van Diggelen, J., and Neerincx, M. (2020). Interpretable confidence measures for decision support systems. *International Journal of Human-Computer Studies*, 144:102493.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv.org*.

[Vrandecic and Krötzsch, 2014] Vrandecic, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

[Wallace et al., 2020] Wallace, D., Tramer, F., Jagielski, M., and Herbert-Voss, A. (2020). Does gpt-2 know your phone number? https://bair.berkeley.edu/blog/2020/12/20/lmmem/.

[Wang et al., 1995] Wang, R., Storey, V., and Firth, C. (1995). A framework for analysis of data quality research. *IEEE Transactions on Knowledge and Data Engineering*, 7(4):623–640.

[Weidinger et al., 2021] Weidinger, L., Mellor, J. F. J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., Kenton, Z., Brown, S. M., Hawkins, W. T., Stepleton, T., Biles, C., Birhane, A., Haas, J., Rimell, L., Hendricks, L. A., Isaac, W. S., Legassick, S., Irving, G., and Gabriel, I. (2021). Ethical and social risks of harm from language models. *ArXiv*, abs/2112.04359.

[Winter and Davidson, 2019] Winter, J. S. and Davidson, E. (2019). Governance of artificial intelligence and personal health information. *Digital policy, regulation and governance*.

[Xafis et al., 2019] Xafis, V., Schaefer, G. O., Labude, M. K., Brassington, I., Ballantyne, A., Lim, H. Y., Lipworth, W., Lysaght, T., Stewart, C., Sun, S., Laurie, G. T., and Tai, E. S. (2019). An ethics framework for big data in health and research. *Asian bioethics review*, 11(3):227–254.

[Yang et al., 2020] Yang, Q., Zhang, Y., Dai, W., and Pan, S. J. (2020). *Transfer Learning*. Cambridge University Press.

[Yeom et al., 2018] Yeom, S., Giacomelli, I., Fredrikson, M., and Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282.

[Zhang and Zhang, 2009] Zhang, E. and Zhang, Y. (2009). *Average Precision*, pages 192–193. Springer US, Boston, MA.

A. Appendix

A.1. Additional Tables

Input Prompt c_is	**Tried Negatives**
MASK	250
MASK is a person.	244
My name is MASK.	235
I am named MASK.	240
MASK is an individual human being.	242
Is MASK your name?	245
Is MASK a person?	249
MASK is not a person.	243
My name is not MASK.	241

Table A.1.: Alternative ranking methodology for the expected rank strategy using **BERT Base** with *dev* dataset

Input Prompt c_i	**Tried Negatives**
MASK	252
MASK is a person.	266
My name is MASK.	264
I am named MASK.	254
MASK is an individual human being.	269
Is MASK your name?	266
Is MASK a person?	265
MASK is not a person.	263
My name is not MASK.	277

Table A.2.: Alternative ranking methodology for the expected rank strategy using **BERT Large** with A_{dev} dataset

Model Size	Tried Negatives
base	239
large	258

Table A.3.: Comparison of BERT Base and BERT Large employing an alternative ranking methodology for the expected rank strategy using the A_{dev} dataset

Input Prompt c_i	Avg. Precision: bert-base	Avg. Precision: bert-large
MASK	0.691	0.669
MASK is a person.	0.686	0.66
My name is MASK.	0.701	0.66
I am named MASK.	0.696	0.671
MASK is an individual human being.	0.678	0.66
Is MASK your name?	0.699	0.665
Is MASK a person?	0.667	0.668
MASK is not a person.	0.685	0.664
My name is not MASK.	0.695	0.656

Table A.4.: Privacy attack strategy (B1) verification with A_{test} dataset for bert-base-NER and bert-large-NER

A.2. Software Information

The code for the experiment presented in this thesis can be found in the following GitHub repository: https://github.com/L-Nux/masterThesis_privacyLeakage. This repository contains all the necessary scripts and data files to reproduce the results of the experiment. Below the versions of the main software used in the experiment are listed.

- Python version: 3.9.12
- Python packages:
  - numpy 1.21.0
  - plotly 5.10.0
  - pandas 1.4.2
  - regex 2022.6.2
  - scikit-learn 1.0.2
  - scipy 1.8.0
  - tensorflow 2.8.0
  - tokenizers 0.12.1
  - torch 1.11.0
  - transformers 4.20.1