UNIVERSITÀ
DEGLI STUDI
DI PADOVA

UNIVERSITY OF PADOVA

DEPARTMENT OF MATHEMATICS
Master degree in Data Science

Exploiting Large Language Models to Train Automatic Detectors of Sensitive Data

Supervisor
Professor Alberto Testolin

Co-supervisor
Dott. Dennis Dosso
Siav S.p.A.

Master Candidate
Simone De Renzis
Student ID: 2056102

Academic Year
2022–2023

Abstract

This thesis proposes an automated system designed to identify sensitive data within text documents, aligning with the definitions and regulations outlined in the General Data Protection Regulation (GDPR). It reviews the current state of the art in Personally Identifiable Information (PII) and sensitive data detection, and how machine learning models for Natural Language Processing (NLP) are tailored to perform these tasks. A critical challenge addressed in this work pertains to the acquisition of suitable datasets for the training and evaluation of the proposed system. To overcome this obstacle, we explore the use of Large Language Model (LLM)s to generate synthetic datasets, thus serving as a valuable resource for training classification models. Both proprietary and open-source LLMs are leveraged to investigate the capabilities of local models in document generation. It then presents a comprehensive framework for sensitive data detection, covering six key domains and proposing specific criteria to identify the disclosure of sensitive data, which take into account the context and the domain relevance. To achieve the detection of sensitive data, a variety of models are explored, mainly based on the Transformer architecture (Bidirectional Encoder Representations from Transformers (BERT)), adapted to fulfill tasks of text classification and Named Entity Recognition (NER). It evaluates the performance of the models using fine-grained metrics, and shows that the NER model achieves the best results (90% score) when trained interchangeably on both datasets, also confirming the quality of the dataset generated with the open source LLM.

Contents

Abstract **vii**

List of figures **xi**

List of tables **xiii**

1 Introduction **1**
- 1.1 Siav S.p.A. 1
- 1.2 Sections overview 2

2 Related works **5**
- 2.1 Personal data in GDPR 5
- 2.2 Personal Data detection 6
  - 2.2.1 PII 6
  - 2.2.2 Sensitive data 7
- 2.3 Models for NLP 8
  - 2.3.1 Word embeddings 8
  - 2.3.2 Convolutional Neural Networks 9
  - 2.3.3 Recurrent Neural Networks 10
  - 2.3.4 Transformers 11
- 2.4 Dataset 14
  - 2.4.1 Synthetic dataset generation 15
- 2.5 Large Language Models 16
  - 2.5.1 Generative Pre-trained Transformer (GPT) 16
  - 2.5.2 LLaMa 17
  - 2.5.3 Generation parameters 17
  - 2.5.4 Prompt Engineering 18
  - 2.5.5 Limitations 19

3 Methodologies **21**
- 3.1 Sensitive data definition 21
  - 3.1.1 Topic Classification 22
  - 3.1.2 Topic Classification and PII 22
  - 3.1.3 Context is Relevant 23
  - 3.1.4 Criteria for Sensitive Data Detection 24
- 3.2 Implementation code and libraries 25

3.3 Dataset 26
3.3.1 Document generation 26
3.3.2 Span labeling 29
3.3.3 Wikipedia documents 32
3.3.4 Dataset cleaning 33
3.3.5 Dataset composition 33
3.3.6 Test set 36
3.4 Models 38
3.4.1 Sentence Classification (SC) 38
3.4.2 Sentence Classification with Context (SCC) 40
3.4.3 NER 43
3.4.4 Spacy Spancat 44
3.4.5 BingAI 45
3.5 Evaluation 47
3.5.1 Evaluation metrics overview 47
3.5.2 Binary Evaluation 48
3.5.3 Multiclass Evaluation 49
3.5.4 Example 49

4 Experiments 53
4.1 Results 53
4.1.1 Span-level results 54
4.1.2 Document-level results 56
4.1.3 Resource footprint 58
4.2 Methodological experiments 58
4.2.1 Datasets 59
4.2.2 Models 60
4.2.3 LLMs generation parameters 63

5 Conclusion 67

References 69

Acknowledgments 71

Listing of figures

2.1 Transformers architecture in its main components. 12
2.2 BERT token encodings and sentence separation. 14

4.1 Comparison of the four models on the three metrics (weighted version) in the span task. . . 55
4.2 Highlighted sensitive spans on a document example. 55
4.3 Comparison of the four models on the three metrics (weighted version) in the document task. 57
4.4 Side to side comparison between the NER models trained on BingAI and OpenLLaMa in the span and document tasks, in terms of (weighted) F1 score. 57
4.5 Score (weighted F1) resulting from training NER models on different size datasets, for both the BingAI and OpenLLaMa datasets. 59
4.6 (Weighted) precision, recall and F1 for the NER model with the standard softmax loss (1 - solid lines) and for the NER model with the weighed softmax loss (2 - dotted lines 61
4.7 Training and validation loss on the NER models when trained with the standard softmax loss (4.7a) or with the weighed one (4.7b) 62
4.8 Example of generation of a document with the custom instruction template 64
4.9 Example of generation of a document with the standard instruction template 65
4.10 Example of generation of a document with the custom instructions and temperature = 0.3 . 65
4.11 Example of generation of a document with the custom instruction template and temperature = 1.5 65
4.12 Example of generation of a document in italian 66

Listing of tables

3.1 List of the documents types generated for the training set, for each category, the sensitive information included in the documents 27
3.2 The presence of each type of sensitive data on the BingAI generated dataset. 33
3.3 The presence of each type of sensitive data on the OpenLLaMa generated dataset. 33
3.4 Description of the dataset generated with BingAI. 34
3.5 Description of the dataset generated with OpenLLaMa. 35
3.6 An overview of the document types found within the test set. The classification of these documents is based on the presence of sensitive data, consideration of whether they contain such data, if they are corner cases, and their absence from the training set. 37
3.7 The process of splitting a document into training example including the context. One notable observation from the lower rows of the table is that the phrase *il raffreddore è una malattia molto contagiosa*, when taken by itself, may not appear to be particularly sensitive. However, when considered within its surrounding context of *raffreddore che ti ha colpito la settimana scorsa*, it becomes evident that the reference to the cold is actually directed towards a person named Anna. 42
3.8 This is an example of how a sentence is tokenized by the BERT tokenizer, and assigned a tag for each token. 43

4.1 BingAI dataset: Span-level metrics for each model in the multi-class setting 54
4.2 OpenLLaMa dataset: Span-level metrics for each model in the multi-class setting 54
4.3 BingAI dataset: Span-level metrics for each model in the binary setting. 56
4.4 OpenLlama dataset: Span-level metrics for each model in the binary setting. 56
4.5 BingAI dataset: Document-level metrics for each model. 56
4.6 OpenLlama dataset: Document-level metrics for each model. 56
4.7 Performance on inference (CPU) 58
4.8 Percentage of wikipedia documents over the total document in the (BingAI) dataset 60
4.9 NER model dataset formatting: performance evaluation with different combination of stride and length parameters 60
4.10 Performance evaluation with different sensitive classes weights on the weighted-softmax loss function for the NER model 61

1

Introduction

In today's digital era, safeguarding personal data has become a priority, especially with the implementation of the General Data Protection Regulation (GDPR). Personal data identified by the GDPR can fall under these two categories: Personally Identifiable Information (PII) and Sensitive Data. PII includes information that directly identifies individuals, while Sensitive Data encompasses more personal details like racial background, health records, political beliefs, and so on (as defined in GDPR Articles 4(1), 9, 10) [1].

This thesis explores the creation of an automated system for identifying sensitive data in text documents, and is the focus of the work carried out during my curricular intership at Siav S.p.A. Following is a brief introduction to the company, and then an overview of the content of the thesis.

1.1 Siav S.p.A.

Siav S.p.A. is an information technology company specializing in dematerialization, electronic document management, and digital processes. Founded by Alfieri Voltan in 1990 in Rubano, Padova, the company primarily concentrates on projects for both business and government organizations.

Siav has become a prominent player in the Italian Enterprise Content Management (ECM) sector, holding over a 20% market share. The company offers various software solutions, cloud-based services, and outsourcing options for Electronic Document Management, Digital Protocol, Workflow Management, Electronic Invoicing, and Digital Preservation. Siav operates from offices in Padova, Milano, Genova, Bologna, and Roma, and extends its reach to Switzerland and Romania.

With a client base exceeding 4,000 across the public and private sectors, Siav boasts a list of notable clients, including Adecco, Agenzia ICE, Amadori, Autovie Venete, and others. In 2021, Siav formalized its commitment to responsible and sustainable operations by adopting the status of a Benefit Company, underlining its dedication to various stakeholders.

Siav's software and Business Process Outsourcing services assist organizations in streamlining document management, simplifying processes, and reducing costs. They offer flexible solutions, from in-house to full outsourcing, with specific expertise in Electronic Invoicing, accounts payable management, and Digital Preservation.

Siav invests significantly in Research and Development: they have a software development center and collaborate with academic institutions on innovative projects, such as Automatic Classification and Process Mining.

The company holds certifications such as Microsoft Gold Competency in Application Development, Oracle Partner Network Gold Membership, SAP integration certification, UNI EN ISO 9001, and UNI CEI ISO/IEC 27001. Since 2014, Siav has been an Accredited Digital Preservation Service Provider, holding the Compliance Certification for Digital Preservation services. They are also qualified as a Cloud Service Provider (CSP) and a certified Peppol Access Point, recognized by AgID and the Open Peppol International Organization.

The research discussed in this thesis evolved during my 6-month internship at Siav, where I had the opportunity to work closely with the R&D department, benefiting from collaborative interactions and idea sharing while being actively involved in the project's development.

1.2 Sections overview

The following sections will delve deeply into various aspects of this endeavor, including existing methods, challenges in obtaining datasets, the proposed algorithmic framework, code implementation details, and model performance assessment.

Section 2.1 elaborates on how the GDPR forms the foundation for data protection in the European Union (EU). It sets strict rules for handling personal data and imposes even stricter regulations for processing sensitive data.

Section 2.2 summarizes the research landscape regarding PII [2] and sensitive data detection. It explains the techniques used in academia and industry [3], highlighting advancements in automated PII detection. It also points out the limited research on sensitive data detection, a gap this paper aims to address comprehensively by building on approaches proposed by other studies [4] [5].

Section 2.3 provides an overview of important machine learning models, offering a background on various techniques considered for the solution [6] [7] [8]. It delves into Large Language Model (LLM)s [9], a significant advancement in Natural Language Processing (NLP), and discusses prompt engineering as a way to interact and get the best results from these models [10] [11] [12].

Acquiring suitable datasets for training and evaluating sensitive data detection models is a significant challenge. Section 2.4 examines the scarcity of appropriate datasets and the use of scraped social media data [13] [14] [15] or costly expert-curated datasets [5]. It then proposes using LLMs to generate synthetic datasets, comparing them with datasets proposed by other researchers. This section emphasizes the flexibility of LLMs in generating various data types when real-world data is unavailable or expensive [16] [17].

Section 3.1 outlines a comprehensive framework for sensitive data detection, categorizing it into six key domains: Health, Political, Sexuality, Judicial, Philosophy, and Ethnic. It defines four critical criteria for effective detection, emphasizing context and sensitivity spans for robust results.

Technical details in Section 3.2 cover the choice of HuggingFace and libraries for running LLMs. It highlights the use of pretrained Bidirectional Encoder Representations from Transformers (BERT) [18] models in the PyTorch framework and introduces OpenLLaMa [19] as an open-source alternative to the Generative Pre-trained Transformer (GPT)-family models.

Section 3.3 explains the process of generating synthetic datasets using LLMs, emphasizing the technical details that are used to optimize the process of document generation and labeling, both with a proprietary LLM based on OpenAI GPT (BingAI), and with an open-source LLM (OpenLLaMa). It is shown how including document from external sources like Wikipedia increases the model robustness, and deals with the preprocessing steps to clean and prepare the datasets.

In Section 3.4, we explore the models employed for sensitive data detection, primarily focusing on the use of BERT models. Our exploration encompasses various approaches, including adapting BERT for text classification tasks with the addition of contextual information, or for Named Entity Recognition (NER) tasks. Furthermore, our experiments extended to testing the Spacy framework, which features built-in NLP task algorithm, as well as an architecture based on Convolutional Neural Network (CNN)s.

Section 3.5 presents a fine-grained evaluation approach, particularly concerning spans, drawing inspiration from commonly used NER metrics [20]. It defines key evaluation metrics such as precision, recall, and F1-score, and discusses different ways of averaging metrics for multiclass evaluation.

Section 4 details experiments that comprehensively evaluate the performance of different models for sensitive data detection on the test set. It confirms the NER model's consistent superiority, achieving approximately 90% accuracy. A comparison is conducted between models trained on datasets generated using BingAI and OpenLLaMa. The previously mentioned results remain consistent across both datasets, with a minor performance decrease observed in the latter. However, this reduction in performance is offset by the privacy-conscious environment and commercially permissive license associated with OpenLLaMa. This section also offers insights into the experiments that influenced the methodology, presenting results related to different techniques and parameter adjustments.

2

Related works

2.1 Personal data in GDPR

The GDPR is a regulation in EU law that aims to protect the privacy and rights of individuals in the EU and the European Economic Area (EEA) with regard to the processing of their personal data[1]. It also regulates the transfer of personal data outside the EU and EEA areas. The GDPR applies to any organization that collects, processes or transfers personal data of individuals in the EU or EEA, regardless of where the organization is located. The GDPR was adopted in 2016 and became enforceable in 2018, replacing the previous Data Protection Directive 95/46/EC.

In the GDPR we can find these articles related to personal data:

- **Art. 4**: *(1) 'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person.*

- **Art. 9**: *(1) Processing of personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation shall be prohibited.*

- **Art. 10**: *Processing of personal data relating to criminal convictions and offences or related security measures based on Article 6(1) shall be carried out only under the control of official authority or when the processing is authorised by Union or Member State law providing for appropriate safeguards for the rights and freedoms of data subjects. Any comprehensive register of criminal convictions shall be kept only under the control of official authority.*

Then, article 4 further specify what falls under the scope of health data. In particular:

- (13) *'genetic data' means personal data relating to the inherited or acquired genetic characteristics of a natural person which give unique information about the physiology or the health of that natural person and which result, in particular, from an analysis of a biological sample from the natural person in question;*
- (14) *'biometric data' means personal data resulting from specific technical processing relating to the physical, physiological or behavioural characteristics of a natural person, which allow or confirm the unique identification of that natural person, such as facial images or dactyloscopic data;*
- (15) *'data concerning health' means personal data related to the physical or mental health of a natural person, including the provision of health care services, which reveal information about his or her health status;*

There are two main categories of personal data that can be identified based on the provided articles. The first category, as described in Article 4(1), includes information that can directly lead to the identification of an individual. This data is commonly known as PII.

In contrast, Articles 9 and 10 refer to a broader range of information that discloses sensitive aspects of a person's life. This includes details such as racial and ethnic origin, political opinions, religious or philosophical beliefs, health data, data related to a person's sex life or sexual orientation, as well as data concerning criminal convictions and offenses.

The GDPR outlines specific criteria that must be followed by data controllers and processors when dealing with personal data. However, the processing of sensitive data, as described in Articles 9 and 10, is subject to even stricter regulations due to the highly sensitive nature of the information it encompasses. For the purposes of this work, we will refer to this category as *sensitive data*, and it is the primary focus of our investigation.

2.2 Personal Data detection

Below is an overview of the most commonly employed techniques for detecting personal data, both in the PII setting and in the broader sensitive data framework.

2.2.1 PII

Detecting PII has been extensively studied in both academia and industry settings. Various automated methods have been developed to identify PII, including rule-based approaches using regular expressions, knowledge-based methods utilizing gazetteers, and NLP methods leveraging NER techniques.

For instance, *Paccosi and Approsio* [2] address the challenge of extracting personal data from Public Administration (PA) documents with an automated approach that involves extracting named entities and semantic relations. To tackle these challenges, the authors propose a NLP approach that combines three techniques: NER, structured-entity identification, and relation extraction. NER entails identifying and classifying named entities mentioned in the documents, such as persons, locations, organizations, laws, and protocols. The authors employ a combination of rule-based and machine learning-based approaches for NER, including the utilization of gazetteers from knowledge bases.

Structured-entity identification focuses on extracting entities that can be recognized without training data, such as dates, times, numbers, email addresses, and the Italian "codice fiscale" (tax identification numbers). Relation extraction aims to identify semantic relationships between entities, such as birth locations, roles in companies, and associations with personal data.

Among the aviable tools to perform PII detection it is worth mentioning *Microsoft Presidio* [3], which is an open-source library, provided by Microsoft, that is designed to identify and anonymize private and personal data within text, including credit card numbers, names, locations, social security numbers, and more.

Presidio emphasizes extensibility and customizability, allowing businesses to tailor the tool to their specific needs. Users have the flexibility to create new or updated PII recognizers, adapt Presidio to support additional languages, utilize state-of-the-art NER models, incorporate new types of anonymizers, and build PII analysis and anonymization pipelines on different platforms using Docker and Python.

2.2.2 Sensitive data

The task of identifying sensitive data is less explored. It is also a more challenging task as it involves techniques that need to deal with a broader and more generic set of expressions.

Petrolini et al. [4] proposed an approach that focuses on four specific categories of sensitive data: politics, health, religion, and sexuality.

The authors established two criteria for determining whether a text contains sensitive data. Firstly, it must include a "sensitive topic". Secondly, the text should contain personal data that can be linked to the sensitive topic mentioned. Identifying personal data and associating it with the corresponding sensitive topic is not always straightforward. Sometimes, personal data may be mentioned separately from the related sensitive topic, or the personal data might be stored in metadata rather than the document text itself, such as sender information in an email. Therefore, accurately identifying personal data and verifying its connection to the sensitive topic throughout the document is necessary, but it carries the risk of generating false positives if not done accurately.

Our approach expands on this by exploring all six sensitive categories mentioned in both Article 9 and 10 of the GDPR. For a complete list and description of these categories, please refer to section 3.1.

As for their approach on detecting sensitive topics, a problem arises when the PII detected and the sensitive topic are not actually linked in a way that discloses sensitive data about the individual. Section 3.1.2 provides a more detailed analysis of this case, and our approach aims to make the detection more robust by feeding the entire document to the classifier. This allows the classifier to capture the entire context of the relationship between the PII and the sensitive topic.

Another approach proposed by *Gambarelli et al.* [5] focuses on building a dataset that comprises sensitive sentences from various categories, ranging from a macro-level of six categories (one of which aligns with the aggregate of our six sensitive categories) to a fine-grained subdivision into over 60 classes. Their approach also considers the context in which words are used, assigning different levels of sensitivity to words based on their surrounding context. The paper also introduces *adversarial sentences*, which contain the same linguistic patterns as sensitive sentences but in a non-sensitive context. These adversarial sentences challenge the models and test their ability to discern the sensitivity of a given word.

Both of these works address the problem at a sentence level, whereas our approach aims to go beyond that and focus on the document level. By considering the entire document as the context for sensitive spans, which may or may not be sensitive based on the words surrounding them, our approach provides a more comprehensive understanding. We explain our approach in more detail in section 3.1.3. By operating at the document level, our model can also better simulate real-world usage scenarios and is expected to achieve better performance compared to aggregating predictions on split sentences.

2.3 Models for NLP

NLP is a specialized field in computer science and artificial intelligence that aims to enable machines to understand, interpret, and generate human language. It involves a wide set of tasks designed to process and analyze text data. Some key tasks include text classification, sentiment analysis, NER, machine translation, language modeling, question-answering systems and text summarization.

We focus on text classification and NER, as they are the tasks we are required to perform to resolve the task at hand.

- Classification is a machine learning task that aims at building a model that can accurately assign input data points to predefined categories or classes. It is a type of supervised learning, meaning the algorithm is trained on labeled data, where each data point is associated with a corresponding class label. During the training process, the algorithm learns patterns and relationships in the input features to make predictions on new, unseen data. The output of a classification model is typically a class label or a probability distribution over class labels, indicating the likelihood of each class for a given input. Common algorithms for classification include logistic regression, Support Vector Machine (SVM), decision trees, random forests, and deep learning methods like neural networks.
- NER focuses on identifying and classifying named entities within a given text. Named entities are real-world objects with specific names, such as names of people, organizations, locations, dates, and numerical values like percentages and monetary amounts.

What follows is a quick overview of the techinques used to perform NLP tasks.

2.3.1 Word embeddings

Word embeddings are a way of representing words as vectors of real numbers that capture their semantic and syntactic similarities [6]. Word embeddings are learned from large corpora of text using various methods, such as neural networks, matrix factorization, or probabilistic models. Word embeddings can be used as input features for text classification and other natural language processing tasks, because they can reduce the dimensionality and sparsity of the data, and also encode meaningful information about the words and their contexts.

There are different types of word embeddings, depending on how they are obtained and how they represent the words. Some of the main types are:

- **Frequency-based word embeddings**: These are word embeddings that are based on counting the occurrence of words in a corpus or a document. Some examples of frequency-based word embeddings are one-hot encoding, count vectorizer, bag-of-words, n-grams, and TF-IDF. These word embeddings are simple and fast to compute, but they tend to ignore the semantics and order of the words, and they suffer from high dimensionality and sparsity.
- **Neural word embeddings**: These are word embeddings that are based on predicting the context or the target word given a word or a sequence of words. Some examples of prediction-based word embeddings are Word2Vec, GloVe, and fastText. These word embeddings are more complex and computationally intensive, but they tend to capture the semantics and syntax of the words, and they have lower dimensionality and sparsity.
- **Contextualized word embeddings**: These are word embeddings that are based on modeling the meaning of a word depending on its context in a sentence or a document. Some examples of contextualized word embeddings are ELMo, BERT, and GPT. These word embeddings are very advanced and powerful, but they require large amounts of data and resources to train and use. They can capture the nuances and variations of the meaning of a word across different contexts.

Word embeddings can be used for text classification by feeding them as input features to a classifier model, which can learn how to map the word vectors to the corresponding labels or categories of the text documents

2.3.2 Convolutional Neural Networks

Despite being created to perform image-related tasks, CNNs have been found to work well on textual data. The architecture is built on top of word embeddings as outlined before: it is a lightweight and effective technique often used for text classification, and it is one of the main models that powers the Spacy framework.

CNNs are a type of artificial neural networks that can process sequential data such as text by applying filters that convolve over the input and extract local features [7]. A typical CNN architecture for text classification consists of four main components: an embedding layer, a convolutional layer, a pooling layer, and a fully connected layer.

- **Embedding layer**: This layer transforms each word in the input text into a vector representation using word embeddings. The embedding layer can use pre-trained word embeddings or learn task-specific word embeddings during training. The output of this layer is a matrix that represents the input text as a sequence of word vectors.
- **Convolutional layer**: This layer applies one or more filters to the output of the embedding layer, sliding over the word vectors and producing feature maps. Each filter can capture a different aspect or pattern of the input text, such as n-grams, word order, or syntax. The output of this layer is a matrix that represents the extracted features from the input text.

- **Pooling layer**: This layer reduces the size and dimensionality of the output of the convolutional layer by applying an aggregation function, such as max, average, or sum, to each feature map. The pooling layer can capture the most important or salient features from the input text and also reduce the computational complexity of the model. The output of this layer is a vector that represents the pooled features from the input text.

- **Fully connected layer**: This layer takes the output of the pooling layer and passes it through one or more dense layers with non-linear activation functions, such as sigmoid, tanh, or relu. The fully connected layer can learn how to map the pooled features to the corresponding labels or categories of the input text. The output of this layer is a vector that represents the predicted probabilities for each label or category.

CNNs can be trained for text classification by using a loss function that measures the difference between the predicted probabilities and the true labels or categories of the input text, such as cross-entropy loss or hinge loss.

2.3.3 Recurrent Neural Networks

Recurrent Neural Network (RNN)s are a type of neural network models that can process sequential data such as text or speech, by using recurrence mechanisms that allow them to store and update information over time [8].

The architecture of a RNN consists of three main components: an input layer, a hidden layer, and an output layer. The input layer takes the input sequence and transforms it into a sequence of vectors using word embeddings or other methods. The hidden layer consists of one or more recurrent units that take the input vectors and the previous hidden states and produce the current hidden states. The output layer takes the current hidden states and produces the output vectors or predictions. The recurrent units can have different structures and functions, such as simple RNNs, Gated Recurrent Unit (GRU)s, or Long Short-Term Memory (LSTM)s.

- **Simple RNN**: This is the basic type of recurrent unit that uses a simple activation function, such as tanh or sigmoid, to compute the hidden state based on the input vector and the previous hidden state. The simple RNN can capture short-term dependencies between the input and output elements, but it suffers from the vanishing gradient problem, meaning that it cannot learn long-term dependencies effectively.

- **GRU** and **LSTM**: These are types of recurrent units that utilize gated mechanisms to control the flow of information within the unit. GRU incorporates reset and update gates, while LSTM incorporates input, forget, and output gates, along with a memory cell. They both effectively capture short-term and long-term dependencies between input and output elements: the gated mechanisms allows to alleviate the vanishing gradient problem commonly encountered in RNNs.

RNNs can be trained for NLP tasks by using different objectives and loss functions depending on the task type and data format. For example, for language modeling, the objective is to predict the next word in a sequence given the previous words, and the loss function is usually cross-entropy between the predicted words and the true words.

RNNs and LSTMs suffers from some drawbacks, such as:

- **Vanishing gradient problem**: This is a problem that occurs when the gradients of the loss function with respect to the model parameters become very small or zero during the backpropagation process. This makes it difficult or impossible to update the model parameters effectively and learn long-term dependencies between the input and output elements. RNNs are especially prone to this problem because they have to backpropagate through many time steps, which can cause the gradients to decay exponentially. LSTMs can alleviate this problem by using gated mechanisms that can preserve or forget information selectively, but they are not immune to it.

- **Exploding gradient problem**: Conversely, the exploding gradient problem arises when gradients become excessively large or infinite during backpropagation. This leads to parameter divergence and prevents the model from converging. Gradient clipping and regularization techniques can help alleviate this problem.

- **Sequential computation**: This is a limitation that arises from the inherent nature of RNNs and LSTMs, which have to process the input sequence one element at a time. This makes them slow and inefficient in terms of training and inference time, especially for long or high-resolution inputs. This also prevents them from taking advantage of parallelization techniques that can speed up the computation on modern hardware devices such as GPUs or TPUs.

For such reasons, Transformer models, described in the following section, are usually preferred.

2.3.4 Transformers

Transformers are a type of neural network models that can process sequential data such as text, speech, or images, by using self-attention mechanisms that allow them to capture the dependencies and relationships between the input and output elements [21]. Transformers can be used for various NLP tasks, such as machine translation, text summarization, text generation, question answering, natural language understanding, and natural language inference.

As shown in Figure 2.1, the architecture of a Transformer consists of two main components: an encoder and a decoder. The encoder takes the input sequence and transforms it into a sequence of hidden representations or embeddings. The decoder takes the encoder output and generates the output sequence. Both the encoder and the decoder are composed of multiple layers of sub-modules, such as:

- **Self-attention layer**: This layer computes the relevance or similarity of each element in the input sequence to every other element, by using a scaled dot-product attention function. The self-attention layer can learn how to focus on the most important or relevant parts of the input sequence for the task at hand. The self-attention layer can also be multi-headed, meaning that it can use multiple parallel attention functions to capture different aspects or perspectives of the input sequence.

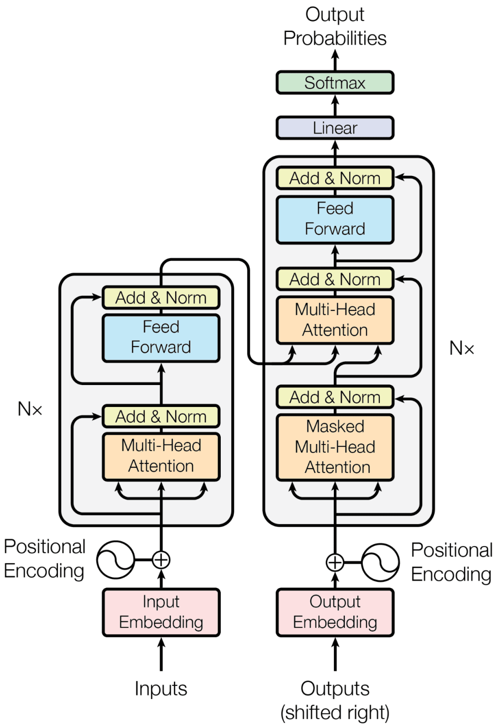


Figure 2.1: Transformers architecture in its main components.

- **Feed-forward layer**: This layer applies a linear transformation followed by a non-linear activation function to each element in the input sequence independently. The feed-forward layer can learn how to map the input elements to more complex or abstract features.
- **Residual connection**: This is a shortcut connection that adds the input of each sub-module to its output, before applying a layer normalization function. The residual connection can help to avoid the vanishing gradient problem and improve the information flow within the model.
- **Positional encoding**: This is a way of injecting some information about the relative or absolute position of each element in the input sequence into the model, since the self-attention layer does not have any inherent notion of order or position. The positional encoding can be learned or fixed, and it can use different functions such as sinusoidal waves or learned embeddings.

In *Karl and Scherp* [22] a comparative investigation is peformed to evaluate the performance of various methods in the domain of short text classification. The study focused on the classification of short texts, such as tweets and product descriptions, which are characterized by limited contextual information and less rigid syntactic structures. The authors compared four pre-trained Transformer models against Bag of Words models, graph-based models, CNNs and LSTMs. The results of the experiments indicated that Transformer models consistently outperformed all other models across both benchmark and real-world datasets in terms of accuracy, demonstrating their efficacy as classifiers for short texts. Moreover, Transformer models exhibited reduced runtime compared to alternative neural network-based models.

For our objective we will focus on Transformers models as they are shown to provide the most effective results in terms of accuracy and speed in the short-text classification domain. In particular, we will adopt the BERT model, which is based on the Transformer architecture.

BERT

BERT stands for Bidirectional Encoder Representations from Transformers [18], one of its main features is bidirectionality, meaning that it can process the input sequence from both left to right and right to left. This allows it to encode the full context of each element in the input sequence, and not just the preceding or following elements. This is achieved by using a special token [MASK] to replace some of the input elements during pre-training, and then predicting the original elements based on their context. This is called the masked language modeling objective.

BERT is pre-trained on a large corpus of text, such as Wikipedia and BookCorpus, using two objectives: masked language modeling and next sentence prediction. The masked language modeling objective is explained above. The next sentence prediction objective is to predict whether two sentences are consecutive or not in the original text, given their embeddings. This objective helps BERT to learn the relationship between sentences. The two sententences are separated by the special token [SEP] (for separator) and the special token [CLS] (for classify) is placed at the beginning of each sentence. Image 2.2 shows an example of this. The embedding of the [CLS] token, which acts as an embedding of the whole sentence, is then passed to a binary classifier. We will reuse the principle of using the [SEP] token to fine-tune our classi-

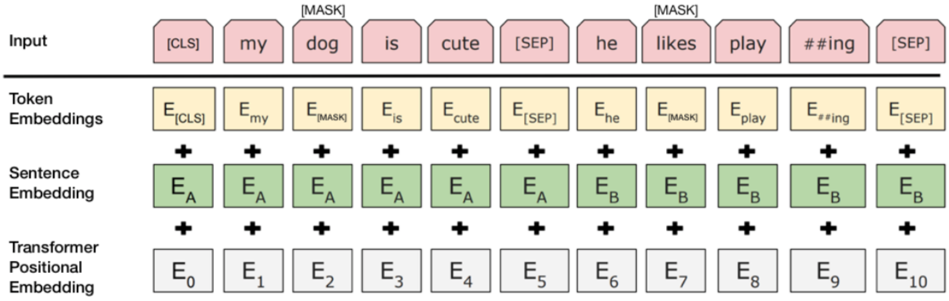


Figure 2.2: BERT token encodings and sentence separation.

fication model to input the model with both the sentence to classify and the contextual text in which the sentence is placed in, as detailed in Section 3.4.2.

BERT can be fine-tuned for specific tasks by adding a task-specific layer on top of the pre-trained model, and then training the whole model on the task data. We will detail the specific layers used in our models in section 3.4

As for the models used for inference, both *Petrolini et al.* [4] and *Gambarelli et al.* [5] used BERT-based models, stacked in a multistep inference pipeline in which a first prediction is done to distinguish between sensitive and non sensitive sentences, and then a finer inference is done to classify the category of the sensitive sentence. This approach requires multiple models to be loaded and kept in memory, and multiple inference steps for every sentence.

Our approach uses a single model for prediction that discriminates between six sensitive categories and a non sensitive one. This speeds up the process of inference as a single step provides the final answer, and a single model is required to be loaded.

2.4 Dataset

The task at hand faces a significant challenge due to the scarcity of available datasets suitable for training a model capable of detecting sensitive data. The sensitive nature of the topics makes it difficult to locate real-world documents containing such data. Moreover, companies are generally hesitant to grant access to private and sensitive documents due to concerns regarding proper data handling protocols.

One publicly available dataset, the *Enron corpus* [15], comprises 619,446 email texts from 158 Enron Corporation users in the period leading up to the company's collapse in December 2001. This corpus was generated by the Federal Energy Regulatory Commission (FERC) during its investigation into Enron's activities.

Another effort focuses on detecting privacy leaks in Tweets [13]. The study aims to quantify and characterize privacy breaches on Twitter, where users inadvertently disclose sensitive information that could have adverse consequences, such as burglaries, legal issues, or discrimination. The authors concentrate on three types of exposure: sharing vacation plans (VP), tweeting under the influence of alcohol (DUI), and revealing medical conditions (MC). They collect a sample of 1.5 million public tweets and manually classify them based on the presence of privacy breaches.

Regarding health data, a dataset has been created for NER tasks to detect health-related keywords such as Name, Surname, Address, Phone, Profession, Department, Doctor, and Patient [14].

One approach, employed by *Petrolini et al.* [4], involves extracting conversations from specific subsections of the Reddit forum that deal with sensitive topics. The sentences extracted are manually labeled with their respective sensitive data category.

Gambarelli et al. [5] constructed two datasets encompassing various sensitive data categories, along with additional categories. These datasets underwent manual labeling, with a subset of sentences curated by domain experts.

Datasets obtained through scraped tweets or Reddit messages offer a cost-effective and convenient means of obtaining data, using the specific categories in which these posts are located to facilitate automatic labeling. However, these datasets may still necessitate some manual effort for more fine-grained labeling, and their homogeneity may limit their effectiveness for training models across diverse document types.

Manually created and curated datasets are undoubtedly of higher quality but are more expensive due to the need for manual labeling, often requiring domain experts' involvement, which incurs considerable costs.

Our proposal involves leveraging the generation capabilities of recent LLM architectures to generate documents and perform automatic labeling, significantly reducing data acquisition costs. Though this approach yields data of lower quality compared to expert-curated datasets, it offers larger quantities of data. As a reference, our dataset (the OpenLLaMa generated one), if split at a sentence level comprises 26'821 data points, exceeding the dataset proposed by *Gambarelli et al.* [5], which contains 5'562 sentences in its fine-grained version. LLMs allows for flexibility in obtaining various types of data, including entire documents of different kinds. Further details about the document types present in our dataset are described in Section 3.3.

Also, employing real-world datasets for training machine learning models can lead to potential security concerns, such as sensitive data extraction from the classifier itself [23] or the possibility of inferring information from neural networks through statistical analysis, as described in [24].

2.4.1 Synthetic dataset generation

Synthetic datasets are a useful alternative to real world datasets, when their aviability or cost makes their use not feasible. Generating datasets with LLMs is an emerging technique that provides good results and allows to cut down the cost of data acquisition.

A research study has been proposed to investigate the cost-effectiveness of using GPT-3 for data labeling in various NLP tasks [16]. The authors employ GPT-3 to generate pseudo-labels for different NLP tasks, such as text classification, question answering, and summarization. These pseudo-labels are then used to train other NLP models, and their performance is evaluated on benchmark and real-world datasets. The authors compare the cost of data annotation using GPT-3 with that of human annotators, measuring the number of tokens required to achieve similar accuracy.

This methodology proposes leveraging GPT-3 to generate pseudo-labels for tasks using few-shot or zero-shot learning. These labels are then used to train task-specific models through supervised or semi-supervised learning. The performance of these models is compared with models trained on human-annotated data or with GPT-3 itself, offering insights into GPT-3's potential for data augmentation and its effectiveness in various tasks and datasets.

Results show that using GPT-3 as a data annotator can lead to cost savings ranging from 50% to 96% compared to using human annotators, depending on the specific task and dataset. Additionally, the models trained on the labels generated by GPT-3 outperform GPT-3's few-shot model in the same task.

Another study introduces a method for generating synthetic data to summarize medical dialogues between physicians and patients [17]. The summaries must be coherent and capture all medically relevant information present in the dialogue. However, acquiring large amounts of labeled data to train effective summarization models is challenging, especially in the medical domain. The proposed method leverages GPT-3 to create synthetic data, with particular emphasis on capturing medically relevant information.

This approach involves a five-step process for creating synthetic medical dialogue data and training a summary model using GPT-3 and BERT. The first step is to prepare a small manually labeled dataset containing medical dialogues and corresponding snippet-level summaries. Then, GPT-3 is used to generate new medical dialogues by providing medically relevant prompts. Next, GPT-3 generates snippet-level summaries for the newly generated dialogues. Additionally, global-level summaries are generated for the new medical dialogues using the dialogue and snippet-level summaries. Finally, the synthetic data is used to train a summary model based on BERT, aiming to improve the model's performance on medical dialogues.

The authors demonstrate that their method achieves comparable results to models trained on 6400 manually labeled examples, using only 210 manually labeled examples and employing few-shot learning and an ensemble method (30 times data reduction). Furthermore, they show that their method produces high-quality synthetic data that, when combined with manually labeled data, yields summaries that are significantly superior in terms of medical accuracy and coherence compared to models trained solely on human data.

2.5 Large Language Models

LLMs are neural networks that are trained on massive amounts of text data to learn the statistical patterns of natural language. They can be used for various natural language processing tasks, such as text generation, text summarization, machine translation, question answering, and conversational agents. As a note, all the information and results discussed in this context pertain to the period up until June 2023. Given the highly dynamic nature of this field, characterized by very active research and development, the reader must acknowledge that the most recent developments may not have been available during the experimentation and writing phases of this work.

2.5.1 GPT

One of the most influential large language models is GPT, which was introduced by OpenAI in 2018 [9]. GPT is a transformer-based model that uses a decoder-only architecture and is pre-trained on a large corpus of web text using the objective of causal language modeling, which predicts the next word in a sequence given the previous words. It can generate coherent and diverse texts on various topics and domains, given a prompt or a query. GPT has been followed by several improved versions: GPT-2, GPT-3, GPT-3.5 and lately GPT-4.

GPT-3, released in 2020, scaled up the model size to 175 billion parameters that can perform zero-shot or few-shot learning on various natural language understanding and generation tasks, such as text summarization, machine translation, question answering, and conversational agents [25]. It shows advanced generation capabilities and language understanding.

In 2023 GPT-4 was released, with a much larger model size and training data than GPT-3 [26]. GPT-4 is likely to be trained on more than 10 trillions of tokens, and composed of more than one trillion parameters, giving it a broader knowledge base and improved contextual understanding. GPT-4 exhibits human-level performance on various professional and academic benchmarks: It passes a simulated version of the Uniform Bar Examination with a score in the top 10% of test takers; it also performs exceptionally well in exams related to language, such as the LSAT (88th percentile), SAT Evidence-Based Reading and Writing (93rd percentile), GRE Verbal (99th percentile). It also demonstrates strong language-based skills in the USABO Semifinal Exam (87/150, 99th - 100th percentile).

ChatGPT is a family of conversational models that are fine-tuned on human-human dialogues from various sources: it poses as a free conversational interface to interact with GPT models on cloud. Pay-per-use Application Programming Interface (API)s are also aviable for intensive usage. The Microsoft search engine Bing integrates GPT models in a free chat interface that can assist in web searches: in one of its configurations, the *Creative* mode, it is powered by GPT-4.

2.5.2 LLaMa

Attempts to reproduce the GPT architecture and training has been proposed: due to its enormous size and the multi-million dollars resources in terms of computing power needed to train such models, big companies are more likely to be able to perform such processes. In particular, a promising attempt is being carried out by Meta, that with its model, LLaMA, achieved results comparable to the ones of GPT-3 [27].

Notably, unlike OpenAI GPT-3, Meta made the trained weights of LLaMa aviable to researchers upon request. Then, due to a leak occurred in February 2023, LLaMa weights were released to the public and became the foundation of open source development on LLMs. Due to the limited size of some of the models, for example the 7b (billions of parameters) and 13b architectures, they are suitable to be run for inference and even fine-tuned on local (high-end) consumer hardware. These models are still not licensed for commercial purposes [*].

2.5.3 Generation parameters

LLMs are trained on large corpus of text to predict the next most probable word or token in a sequence, based on the previous context. The context is composed by the system message and the user prompt, the first one being a general message that specifies the task at hand (for example, answering user questions), the

---

[*] `https://github.com/facebookresearch/llama/blob/main/LICENSE` (Last visited: June 2023)

user prompt consisting on the specific instruction the LLMs is asked to solve. LLMs are usually fine-tuned in order to be adapted from text-completion models to instruct models, so that they are able to follow and adapt to a various range of users' requests.

LLMs are calibrated with some parameters that affect the quality and diversity of the generated text [28]. Some of the main generation parameters are:

- **Temperature**: A parameter that controls the randomness of the sampling process. A higher temperature means more diversity and creativity, but also more errors and inconsistency. A lower temperature means more coherence and accuracy, but also more repetition and blandness.
- **Top-k**: A parameter that controls the number of tokens that are considered for sampling. It only keeps the k most likely tokens, and discards the rest. This reduces the diversity and randomness of the sampling, but also avoids sampling very rare or irrelevant tokens.
- **Top-p**: A parameter that controls the probability mass of the tokens that are sampled. It filters out tokens whose cumulative probability exceeds a certain threshold, and renormalizes the remaining tokens.
- **Typical-p**: A parameter that controls the typicality of the generated tokens. It penalizes tokens that have a low probability given the previous context, and boosts tokens that have a high probability. This encourages the model to generate more typical and expected text, rather than surprising or novel text.
- **Epsilon cutoff**: A parameter that controls the minimum probability of a token to be sampled. It filters out tokens whose probability is lower than a certain threshold, and renormalizes the remaining tokens. This avoids sampling very unlikely or erroneous tokens, but also reduces the diversity and creativity of the sampling.
- **Repetition penalty**: A parameter that controls the repetition of tokens or n-grams in the generated text. It penalizes tokens that have already appeared in the previous context, and boosts tokens that have not. This discourages the model from generating redundant or bland text, but also allows for some repetition when necessary.

2.5.4 Prompt Engineering

One of the aspects that plays an important role in obtaining a insightful response from LLMs is the quality of the request they are prompted with. "Prompt engineering" is a term used to describe the techniques involved in crafting carefully constructed instructions or user prompts that guide the language model's generation process towards producing relevant and accurate responses. By tailoring prompts to suit the task requirements and context, prompt engineering enables language models to better understand and follow a wide range of user requests. It also plays a crucial role in achieving more controlled and desirable outputs by utilizing various techniques, such as context framing, explicit instruction, or even providing examples.

One effective technique in prompt engineering is the concept of "Chain-of-Thought." The techniques entails a series of intermediate reasoning steps the LLM is asked to perform, by providing questions and answers with intermediate explanations that illustrate the reasoning process. By utilizing a chain of thought, models can focus their attention on relevant facts, avoid distractions, and maintain logical coherence: empirical evidence shows that this approach improves performance across a range of tasks involving arithmetic, common-sense, and symbolic reasoning. The observed gains in model performance can be remarkably substantial [10].

A research study aims to investigate the task of job classification in a real-world context, specifically determining whether an English language job advertisement is suitable for a graduate or entry-level position. The study compares this task using LLMs in both few-shot and zero-shot classification contexts. The results demonstrate that with a well-designed prompt, a zero-shot classifier using GPT-3.5 outperforms all other models, achieving a 6% increase in Precision and Recall compared to the best supervised approach. Furthermore, the study observes that the formulation of the prompt plays a critical role in eliciting appropriate "reasoning" in the model, and seemingly minor aspects of the prompt significantly influence the model's performance [11].

Some general guidelines are empirically found to be useful when crafting the instruction prompts. Most of them come from common sense principles of clear communication, such as:

- Use clear and concise language to communicate the intent and context of the prompt.
- Provide specific and relevant information to constrain or customize the output and interaction.
- Improve the quality and efficiency of the output and interaction by refining the prompt or suggesting alternatives. For example, prompts that start with *Act as an expert on [specific domain]...* seem to improve the focus and insighfulness of answers.
- Leverage existing knowledge or notation to express ideas that are hard or verbose to describe in natural language.
- Provide some examples to guide the generation towards a desired format of answer and a better understanding of the objective [12].

2.5.5 Limitations

Despite being one of the most relevant advancement in the AI domain, LLMs pose some limitations [29]:

- **Bias**: LLMs can unintentionally demonstrate bias due to biased training data, biased user interactions, algorithmic bias, and contextual bias. These biases can lead to discriminatory or inaccurate responses.
- **Information Hallucination**: LLMs may generate information not based on their training data, leading to factually incorrect or fictional outputs. Hallucinations are a complex problem in LLMs and can be addressed through better training methods and datasets.
- **Explainability**: LLMs' large parameter count makes them difficult to explain and understand. The lack of transparency hinders insights into their decision-making process, raising concerns about accountability and trust.

- **Reasoning Errors**: LLMs can make mistakes in logical reasoning, particularly in complex logical operations or ambiguous prompts.
- **Challenges in Specific Tasks**: LLMs may struggle with tasks like identifying and correcting spelling errors due to their statistical nature.
- **Adversarial Attacks**: LLMs can be susceptible to various adversarial attacks, including misleading prompts, 'jailbreak' attacks, and data poisoning strategies.
- **Environmental Impact**: The training process for LLMs requires substantial energy, leading to greenhouse gas emissions: this has significant environmental implications, consuming large amounts of water and electricity. This raises concerns about sustainability and calls for eco-friendly data center practices and renewable energy sources.

Model collapse

Text generated with LLMs is being used to fine tune smaller models, like LLMs themselves. A study investigates model collapse as a concerning phenomenon observed in generative models, including LLMs, when trained on data produced by other models they gradually lose essential information about the true data distribution and ultimately converge to a state of minimal variance and diversity [30]. This degradation occurs in various types of models, such as Gaussian Mixture Models, Variational Autoencoders, and Large Language Models. The underlying causes of model collapse can be attributed to statistical approximation error emerging from finite sampling, leading to the loss of rare events and thereby contributing to the collapse; and from functional approximation error arises from the limited expressiveness of the model or overfitting, further exacerbating the problem.

As we focus on building a classifier model and not a generative one, it is reasonable to argue that the extent of the problem's influence on our specific model may be relatively limited.

3

Methodologies

3.1 Sensitive data definition

In this section, we aim to establish a comprehensive framework for the detection of sensitive data. The focus of our analysis, as outlined in the introduction, centers around six key categories of sensitive data:

- **Health**: This category encompasses information pertaining to the physical and mental well-being of individuals. It includes details regarding existing diagnoses, medical conditions, and disabilities.
- **Political**: The political category involves information related to an individual's political beliefs. This encompasses their political orientation, specific party affiliation, as well as membership in work unions or similar organizations.
- **Sexuality**: Within the realm of sexuality, we consider information concerning an individual's sexual orientation, habits, and gender identity.
- **Judicial**: The judicial category encompasses data associated with legal matters, such as offenses, crimes, charges, pending criminal proceedings, accusations, and trial proceedings involving an individual.
- **Philosophy**: Information regarding an individual's philosophical and religious beliefs and affiliations falls within the scope of the philosophy category.
- **Ethnic**: The ethnic category pertains to an individual's ethnic origin and heritage.

The following section aim at providing some examples to better define the scope of detection of sensitive data.

3.1.1 Topic Classification

One approach to detecting sensitive data involves attempting to classify the topic of a given text. Each of the six categories of data corresponds to a well-defined topic, characterized by specific associated vocabulary.

To illustrate why this approach may not be suitable for our task, let us consider the following document:

```
Avvocato libero professionista e collaboratore presso Studio Legale,
specializzato in ambito civile: recupero crediti, contenzioso bancario,
procedure esecutive, azioni a tutela del patrimonio, diritto condominiale;
in ambito penale: reati contro il patrimonio, reati contro la persona, guida in
stato di ebrezza e stupefacenti.
```

This document describes the specialization of a lawyer, highlighting the specific areas in which the lawyer has expertise. The topic of this document, due to the vocabulary being used, can be easily categorized as “Judicial.” However, upon closer examination of the complete text, it becomes apparent that it does not contain any information related to offenses or crimes committed by individuals.

3.1.2 Topic Classification and PII

To address the challenge highlighted in the previous section, one possible approach is as follows:

1. Detect the topic of a document.
2. If the identified topic is one of the six sensitive topics, search for any instances of PII that pertain to an individual.

The underlying idea is that if a sensitive topic is present and a person is mentioned in the text, there should be some relationship between the sensitive topic and the disclosure of sensitive information about that person. This approach is used in *Petrolini et al.* [4] and has been discussed in 2.2.

While this approach may hold true in certain cases, it is crucial to examine the nature of the relationship between the topic and the person mentioned.

Let us consider the following document:

```
La malattia di Alzheimer-Perusini, detta anche morbo di Alzheimer, demenza
presenile di tipo Alzheimer, demenza degenerativa primaria di tipo Alzheimer
o semplicemente Alzheimer, è la forma più comune di demenza degenerativa
progressivamente invalidante con esordio prevalentemente in età presenile
(oltre i 65 anni).
La patologia è stata descritta per la prima volta nel 1906, dallo psichiatra e
neuropatologo tedesco Alois Alzheimer.
```

In this example, we have a sensitive topic (Health) and a person mentioned: Alois Alzheimer. However, the relationship between these two elements does not disclose health information about Mr. Alois himself; rather, he is the individual who first described the condition.

3.1.3 Context is Relevant

An additional aspect that needs to be considered is the context in which a sentence is situated. The same sentence can be sensitive in one context and non-sensitive in another, despite its identical wording.

To illustrate this point, let us examine two documents:

> Oggetto: Come va?
> Ciao Anna,
>
> come va? Ti scrivo per sapere se ti sei ripresa dal raffreddore che ti ha colpito la settimana scorsa. Mi hai detto che avevi febbre, tosse e mal di gola e che ti sentivi molto stanca. Spero che non sia nulla di grave e che tu abbia seguito le indicazioni del medico.
>
> Ti scrivo anche per dirti che mi dispiace di non essere potuta venire a trovarti, ma sai che in questo periodo sono molto impegnata con il lavoro e la famiglia. Non volevo neanche rischiare di contagiarmi o di contagiare qualcun altro. **Il raffreddore è una malattia molto contagiosa**. Ti chiedo scusa se ti sei sentita trascurata o dimenticata.
>
> Ti mando un bacio e ti auguro di guarire presto. Fammi sapere come stai, mi raccomando.
> La tua amica,
> Sara

and:

> **Il raffreddore è una malattia molto contagiosa**. È un'infezione comune, non grave che, di solito, si risolve entro 5-10 giorni. I disturbi (sintomi) che provoca, comuni per tutti i virus che possono causare il raffreddore, comprendono starnuti, produzione abbondante di muco, naso chiuso o che cola (congestione nasale), catarro e mal di gola, tosse, sensazione di stanchezza. Possono presentarsi anche sintomi più gravi, tra cui una temperatura elevata (febbre), mal di testa e dolori muscolari, anche se sono più tipici dell'influenza.

In both documents, the sentence *Il raffreddore è una malattia molto contagiosa* (cold is a highly contagious disease) is present. However, its sensitivity depends on the context in which it appears.

In the first document, which is an email exchange between two friends, the sentence is clearly related to a specific individual (Anna) without explicitly stating so.

In contrast, the second document provides a general description of the common cold from a Wikipedia page. In this context, the same sentence is not sensitive as it does not pertain to any specific person.

Understanding the relevance of context is critical in the task of span detection, as it enables the distinction of sentences that may share the same wording but vary in their classification based on the context in which they are situated.

3.1.4 Criteria for Sensitive Data Detection

In order to effectively detect sensitive data, the following criteria are followed:

1. **Categorization of Sensitive Data**: Sensitive data is classified into six key categories, namely Health, Political, Sexuality, Judicial, Philosophy, and Ethnic. However, relying solely on topic classification may not be sufficient, as the presence of a sensitive topic does not guarantee the disclosure of sensitive information.

2. **Identification of Topics and PII**: Identifying the topic of a document and searching for instances of PII within that topic is not enough for the detection process: the relationship between the sensitive topic and the disclosure of sensitive information about a person would need to be examined.

3. **Consideration of Context**: Context plays a critical role in determining the sensitivity of a sentence. The same sentence can be sensitive or non-sensitive depending on the context in which it appears. Understanding the relevance of context enables the distinction of sentences that share similar wording but differ in their sensitivity classification.

4. **Detection of Sensitive Spans**: To accurately detect sensitive data, the focus is on identifying spans of text that, within their contextual boundaries, have the potential to disclose sensitive information about individuals. The aim is to remove such spans in a way that, if the text is read without them, it becomes impossible to reconstruct the specific sensitive information.

By adhering to these criteria, an effective sensitive data detection framework can be defined, enabling the identification and protection of sensitive information across various types of texts.

3.2 Implementation code and libraries

Regarding the implementation of the models described in the following sections, we used the following languages and libraries:

- **Python**: We used Python as the programming language for both data preparation and processing, serving as the foundation for all the model implementations.
- **HuggingFace**[*]: We utilized the HuggingFace library, which comprises various steps in NLP task processing. Notably, the `Transformers` library within HuggingFace is a Python package that offers open-source implementations of transformer models for text, image, and audio tasks. It is compatible with deep learning libraries such as PyTorch, TensorFlow, and JAX. We utilized the pretrained BERT models made aviable by this library within the PyTorch framework. Additionally, the HuggingFace hub hosts many significant LLMs.
- **text-generation-webui**[†]: This project provides an interface to locally run large language models (LLMs). It enables users to switch between different models, perform fine-tuning (LoRa), use multi-modal pipelines, and generate text in various modes (e.g., chat, notebook, instruction) while adjusting generation parameters. It offers a Graphical User Interface (GUI) and a comprehensive API interface that can be run as a local or remote endpoint. But its purpose extends beyond these functionalities as it aims to unify the components involved in running LLMs. It can be used with various backends, which are different implementations of large language models capable of generating text based on different inputs and parameters. Among them, we mention:
    - `llama.cpp`: A C++ implementation of LLaMA, a large language model that uses 4-bit quantization to reduce memory usage and increase speed.
    - `AutoGPTQ`: A framework for automatically quantizing GPT models to 4 bits without retraining.
    - `ExLlama`: A more memory-efficient rewrite of the HuggingFace transformers implementation of LLaMA for use with quantized weights. This is the backend we employed due to its superior performance in terms of quality and speed of generation (approximately 20 tokens per second).
- **OpenLLaMa**: This project aims to reproduce the implementation and training of the LLaMa model [19]. It shares the same architecture as LLaMa but it is trained on an open-source dataset, making its license open source and commercially permissive. We opted for the 13b (13 billion parameters) model, which at the moment of writing is the largest available while still runnable on reasonably powered hardware.
    - In particular we used an instructed fine-tuned model as provided by VMware[‡]
    - This fine-tuned version is quantized in 4bit in order for us to be able to perform generation in an environment with 15 GB of VRAM GPU. The released quantized model has been provided by the user TheBloke[§] in the HuggingFace hub.

---

[*] `https://huggingface.co/`

[†] `https://github.com/oobabooga/text-generation-webui`

[‡] `https://huggingface.co/VMware/open-llama-13b-open-instruct`

[§] `https://huggingface.co/TheBloke/open-llama-13b-open-instruct-GPTQ`

3.3 Dataset

As described in section 2.4 the lack of an aviable dataset prompted us to consider the use of generative models, such as LLMs, to generate a synthetic dataset. This process involves two distinct phases: document generation, which entails the creation of documents of specific types, and span labeling, which involves the detection and categorization of sensitive data spans within the generated documents.

For our experimentation, we employed two types of LLMs. The first model, BingAI, described in 2.5, represents one of the most famous and powerful LLM and is powered by GPT-4. However, due to restrictions on its usage for commercial purposes, we also explored the utilization of an open-source local model called OpenLLaMa, which allows for commercial use. The subsequent sections outline the process of document generation and labeling as conducted using both of these models.

Subsequently, we observed how the inclusion of documents from external sources, such as Wikipedia, was useful to improve robustness of the trained models. The rationale for this decision is detailed later in this document.

An overview of the generated dataset is provided, including pertinent details regarding the composition of the datasets in terms of the number of documents, the presence of sensitive data, and their categorization.

Finally, the test set is introduced, which comprises a standardized collection of documents generated by BingAI. This test set is distinct from the training data and is exclusively employed for comparative purposes and benchmarking the performance of various models used for sensitive data detection.

3.3.1 Document generation

LLMs possess the capability to generate text of specific types, including the documents of interest in our study. To proceed with the generation process, we require a list of types of documents that contain sensitive information. These documents can be categorized into two distinct groups:

- Inherently Sensitive Documents: This category encompasses documents that are inherently focused on a specific theme, such as clinical records or crime reports. The entirety of these documents typically revolves around sensitive information, often pertaining to specific individuals.

- Mention-Based Sensitive Documents: This category comprises documents that primarily belong to non-sensitive domains, such as emails, newspaper articles, or Curriculum Vitae. However, these documents may contain certain spans of text that disclose sensitive information regarding individuals.

In order to ensure the dataset's effectiveness in addressing both of these scenarios, it is essential to include samples from both categories.

Table 3.1 is a complete list of the documents for each category of sensitive data. We derived these document types through extensive web searches in order to gather the most diverse range of document types at the best of out capability.

Table 3.1: List of the documents types generated for the training set, for each category, the sensitive information included in the documents

Sensitive data	Document title	Sensitive info
Health	Cartella clinica	salute, disabilità
	Referto psicologico	
	Referto psichiatrico	
	Ricetta medica	
	Articolo di giornale	
	Curriculum Vitae	
	Email	
Politic	Tessera di partito	orientamento politico, appartenenza sindacale
	Iscrizione a sindacato	
	Curriculum Vitae	
	Email	
Sexuality	Cartella clinica	identità sessuale, orientamento sessuale
	Referto psicologico	
	Referto psichiatrico	
	Ricetta medica	
	Articolo di giornale	
	Certificato di unione civile	
	Certificato di matrimonio	
	Email	
Judicial	Certificato penale	accuse, reati o illeciti commessi
	Certificato dei carichi pendenti	
	Certificato penale del casellario	
	Sentenza penale	
	Verbale di arresto	
	Applicazione di misure cautelari	
	Avviso di conclusione delle indagini	
	Decreto penale di condanna	
	Informazione di garanzia	
	Verbale di denuncia	
	Referto psichiatrico	
	Identikit di polizia	
	Articolo di giornale	
	Partecipazione a gara d'appalto	
	Email	
Philosophical	Certificato di battesimo	credo religioso, credo filosofico
	Partecipazione a gruppo religioso	
	Articolo di giornale	
	Email	
	Certificato di matrimonio	
	Dichiarazione	
Ethnic	Email	origine etnica
	Curriculum Vitae	
	Articolo di giornale	
	Certificato di nascita	
	Albero genealogico	
	Analisi della discendenza	
Other	Report finanziario di azienda	
	Guida turistica di una città	
	Recensione di un libro	
	Ricetta di cucina	
	Email tra colleghi	
	Lettera commerciale di azienda	
	Rapporto commerciale di azienda	
	Fattura di azienda	
	Bilancio aziendale	
	Dichiarazione fiscale	
	Circolare interna	
	Catalogo di prodotti o servizi	
	Articolo di scienza	
	Volantino pubblicitario	

It is worth mentioning that the dataset also includes documents without any sensitive information, as indicated in the "Other" row of Table 3.1. These documents were included to provide negative examples within the dataset.

Table 3.1 also specifies the "Sensitive info" to be included into the document. This information improves the quality and alignment of the generated documents. The overall process of document generation is facilitated by an automated script that selects a document title, retrieves the corresponding sensitive information, and concatenates these elements with the appropriate prompt in the following structure:

```
Generate a fictional but realistic document about NAME of the type
“DOCUMENT TITLE”, which includes information about SENSITIVE INFO of
NAME.
```

In the place of DOCUMENT TITLE a document title is selected from the options provided in Table 3.1.

Similarly, a sensitive information category is chosen from the available options in the place of SENSITIVE INFO. Additionally, to enhance document diversity, a randomly selected name from a predefined list is used and incorporated in the place of NAME. This practice has been observed to contribute to the variability of the generated documents.

BingAI

What follows is an example of a prompt used in the BingAI chat interface for document generation:

```
Puoi generare un documento finzionale ma verosimile riferito a Anna
del tipo 'Email', che includa informazioni sulla disabilità di un
dipendente? Usa uno stile vario. Racchiudi il documento in un blocco
di markdown.
```

OpenLLaMa

The OpenLLaMa model, due to its smaller size and limited capabilities, as well as its primary training on English corpora, does not exhibit sufficient proficiency in generating text in languages other than English, including Italian. Consequently, the prompt and the generated document for the OpenLLaMa model are in English and later translated into Italian using translation services.

The document titles to be generated are also translated into English, but they are the same of the ones shown in the Table 3.1.

A sample prompt for the OpenLLaMa model is the following:

```
Please generate a fictional but realistic document referred to Olivia
of the type 'Medical prescription', which explains in great detail
health information about Olivia.
```

One of the advantages of utilizing a local model is the ability to exercise complete control over its parameterization. This includes the customization of text generation parameters and the system prompt, which is typically predefined and fixed in platforms like BingAI Chat.

The document generation phase incorporated the following parameters, whose significance and implications have been elucidated in Section 2.5.3:

'temperature'	0.9	'top_k'	0
'top_p'	0.7	'min_length'	0
'typical_p'	1	'no_repeat_ngram_size'	0
'epsilon_cutoff'	0	'num_beams'	1
'eta_cutoff'	0	'penalty_alpha'	0
'tfs'	1	'length_penalty'	1
'top_a'	0	'early_stopping'	FALSE
'repetition_penalty'	1.1	'mirostat_mode'	0

As for the system prompt, we adopted the Alpaca format of interaction[¶], which is consistent with the structure employed during the fine-tuning phase of the model (instruct).

```
user: “### Instruction:”
bot: “### Response: Sure! Here's the generated document:”
context: “You are a professional document generator. You are asked to generate
a document of a specific type: make it as detailed as possible, making up and
inserting realistic information (names, dates, jobs, occupations...) where
necessary. The generated document must be very long so that it appears more
realistic.”
```

This format consists of a conversational exchange with alternating user instructions (prompt) and model responses. The response, representing the generated text from the model, can be preceded by a predefined phrase, such as *Sure! Here's the generated document:*. By providing an explicit starting point for the model's response, it is guided towards generating text in alignment with the given task, reducing the likelihood of diverging from the intended objective.

Furthermore, the specific context provided to the model plays a significant role. As mentioned in Section 2.5.4, instructing the model to "impersonate" specific roles, such as a *document generator* in this case, has been observed to improve the quality of the responses. This contextual framing enhances the model's ability to generate coherent and realistic text tailored to the designated task.

3.3.2 Span labeling

The subsequent step in constructing the dataset and conducting training involves the labeling of sensitive spans within the generated documents. Labeling is commonly recognized as the most resource-intensive aspect during dataset creation, as it necessitates domain expertise and manual effort to accurately assign labels to the training data. However, leveraging the zero-shot (or few-shot) capabilities of LLMs presents a promising approach to reduce labeling costs.

Our objective is to obtain the starting and ending character indices of each detected sensitive span within a given document, along with the corresponding category of sensitive data. Similar to the document generation process, we will outline the methodologies employed for performing this task with both BingAI and OpenLLaMa models. However, the approach differs between the two LLMs, as each model demonstrated distinct efficacy in addressing this task.

BingAI

For the BingAI model, the chosen approach involves prompting the model to generate the document provided as input, but with the sensitive information spans "censored" or concealed with a specific tag.

To guide the model in detecting specific types of sensitive information, the prompt is automatically constructed based on the known sensitive category data associated with the given document. For instance, if the document contains health-related information, the prompt may take the following form:

[¶] `https://github.com/tloen/alpaca-lora/blob/main/templates/alpaca.json`

```
Puoi censurare tutte e sole le porzioni di frasi che contengono informazioni
o possono ricondursi alla disabilità di Anna? Fornisci il documento con sole
frasi che non hanno niente a che fare con la disabilità di Anna. Leggendo il
documento non devo essere in grado di ricostruire alcun'informazione relativa
alla disabilità di Anna. Usa l'etichetta [DATI_SALUTE] per sostituire le
porzioni di frase che contengono informazioni relative la salute (fisica o
mentale) di Anna.
```

Similar to the prompt for document generation, the labeling prompt also follows a structured format with specific variable words that are filled based on the document type and the associated sensitive information.

```
Oggetto: Come va?
Ciao Anna,

come va? Ti scrivo per sapere se [DATI_SALUTE]. Spero che non sia nulla di
grave e che tu abbia seguito le indicazioni del medico.

Ti scrivo anche per dirti che mi dispiace di non essere potuta venire a
trovarti, ma sai che in questo periodo sono molto impegnata con il lavoro e
la famiglia. [DATI_SALUTE]. Ti chiedo scusa se ti sei sentita trascurata o
dimenticata.

Ti mando un bacio e ti auguro di guarire presto. Fammi sapere come stai, mi
raccomando.
La tua amica,
Sara
```

This is an example of result as returned by the labeling prompt. In this instance, the spans pertaining to Anna's health have been substituted with the tag "[DATI_SALUTE]" (health data). The retrieval of the starting and ending character indices for each span is performed using alignment algorithm based on dynamic programming, inspired by techniques commonly employed in the field of Bioinformatics [31].

OpenLLaMa

Unlike the BingAI model, the OpenLLaMa model presents challenges in adhering to a specific answer structure and comprehending the nuances of sensitive data detection. Consequently, an alternative strategy was devised to simplify the task and facilitate the model in following a more rigid structure.

In this approach, the document is initially divided into sentences. To ensure manageable sentence lengths, shorter sentences are aggregated while longer sentences are truncated. These individual sentences are then presented to the OpenLLaMa model, which is instructed to assign a label indicating whether each sentence is sensitive or non-sensitive.

To aid the model in understanding the task and generating output in the desired format, the prompt is constructed with illustrative examples. This serves as guidance for the model, facilitating its comprehension of the output generation format and the overall task. Consequently, the transition is made from a one-shot learning scenario, as employed with BingAI, to a few-shot learning scenario for OpenLlama.

Similar to the labeling prompt for BingAI, the prompt for OpenLlama is tailored based on the type of sensitive data present in the document for which detection is to be performed.

~~~
Frow the following sentences, classify them as if they contain information about health. If so, put a
label [HEALTH] next to the sentence. If not, put the label [NON_HEALTH] next to the sentence.

For example:
Given these sentences:
```
“Caro Amico”
“Ti scrivo per darti delle novità su Charles, il nostro amico e collega che lavora al dipartimento di
informatica della St. Charles Community College.”
“Bene, ora posso dirti che Charles ha una disabilità psicosociale, cioè una condizione che gli causa
difficoltà nel relazionarsi con gli altri, nel gestire lo stress e nel mantenere la concentrazione.”
“Charles mi ha confidato che soffre di questa disabilità da diversi anni, ma che non ne aveva mai
parlato con nessuno per paura di essere giudicato o discriminato.”
“Il dottore gli ha prescritto dei farmaci per stabilizzare i suoi livelli ematici.”
“Ti prego di non dirlo a nessuno.”
“Charles inoltre soffre di mal di testa e di vertigini”
“Lui non vuole che si sappia della sua condizione di salute”
“Vorrei poterlo aiutare, ma è difficile intervenire in questi casi: a volte le persone non vogliono
essere aiutate.”
“E' difficile gestire i problemi di salute, ma Charles è una persona forte e coraggiosa. ”
```

I classified them if they contain information about health. If so, I put a label [HEALTH] next to
the sentence. If not, I put the label [NON_HEALTH] next to the sentence:
```
“Caro Amico” [NON_HEALTH]
“Ti scrivo per darti delle novità su Charles, il nostro amico e collega che lavora al dipartimento di
informatica della St. Charles Community College.” [NON_HEALTH]
“Bene, ora posso dirti che Charles ha una disabilità psicosociale, cioè una condizione che gli causa
difficoltà nel relazionarsi con gli altri, nel gestire lo stress e nel mantenere la concentrazione.”
[HEALTH]
“Charles mi ha confidato che soffre di questa disabilità da diversi anni, ma che non ne aveva mai
parlato con nessuno per paura di essere giudicato o discriminato.” [HEALTH]
“Il dottore gli ha prescritto dei farmaci per stabilizzare i suoi livelli ematici.” [HEALTH] “Ti
prego di non dirlo a nessuno.” [NON_HEALTH]
“Charles inoltre soffre di mal di testa e di vertigini” [HEALTH]
“Lui non vuole che si sappia della sua condizione di salute” [HEALTH]
“Vorrei poterlo aiutare, ma è difficile intervenire in questi casi: a volte le persone non vogliono
essere aiutate.” [NON_HEALTH]
“E' difficile gestire i problemi di salute, ma Charles è una persona forte e coraggiosa.” [HEALTH]
```

Given these sentences:
```
~~~

The labeling prompt can be divided into two distinct sections: the violet portion consists of example sentences without any labels, while the subsequent blue section presents the same sentences, but with an appended tag at the end of each sentence indicating its sensitivity. The prompt then resumes with the same introductory text as the violet section, followed by the sentences from the tokenized document that need to be labeled, which are appended to the prompt.

To guide the OpenLlama model in adhering to a consistent structure, in the system prompt the answer is set to begin with the same introductory text as the blue section, ensuring that the model follows the prescribed formatting.

```
user: “### Instruction:”
bot: “### Response: I classified them if they contain information about health.
If so, I put a label [HEALTH] next to the sentence. If not, I put the label
[NON_HEALTH] next to the sentence:”
context: “Below is an instruction that describes a task. Write a response that
appropriately completes the request.”
```

It is important to highlight that before undergoing tokenization and labeling, the document is translated from English to Italian using a translation service. No significant disparity in the labeling capabilities between the two languages has been observed.

3.3.3 Wikipedia documents

In addition to the documents containing sensitive information, the generated dataset presented in Table 3.1 includes documents that pertain to non-sensitive topics. However, it has been observed that incorporating a subset of documents related to sensitive categories, yet devoid of any sensitive information about individuals, is highly beneficial. This case is discussed in detail in Section 3.1.1. These documents serve as negative examples, enhancing the model's ability to discern and distinguish sensitive information within different topics.

To achieve this, we have introduced a set of documents sourced from encyclopedic platforms, specifically scraped pages from Wikipedia. These documents are thematically aligned with the sensitive topics, such as health and politics. Importantly, these Wikipedia documents do not disclose any sensitive information about individuals. By including such negative examples in the dataset, the model is encouraged to develop robust discrimination skills between sensitive and non-sensitive information.

The scraping process was conducted utilizing the Wikipedia APIs, which facilitated the search for relevant categories and retrieval of associated pages. By leveraging the hierarchical structure of categories within Wikipedia, we were able to specify the desired depth of results. Deeper levels of depth yielded a higher number of retrieved pages, while shallower levels produced more general-topic pages. We limited the depth to the second level.

Specifically, the following categories were scraped: *Ideologie politiche, Movimenti politici*, *Malattie*, *Diritto penale, Diritto civile*, *Sessualità*, *Filosofia, Religione*.

Only meaningful paragraphs were retained from the scraped content. Consequently, sections such as *Note*, *Voci correlate*, *Altri progetti*, *Collegamenti esterni*, *Fonti*, and *Annotazioni* were excluded from the dataset, as they primarily consist of links and notes, and they lack substantive content.

Furthermore, Biography sections were also removed as they have the potential to resemble the disclosure of sensitive information due to their association with individuals. We aimed to ensure that the paragraphs included in the dataset were solely focused on generic topics rather than specific personal details.

3.3.4 Dataset cleaning

The dataset undergoes a cleaning process prior to being fed into the models. Specifically, all special characters, with the exception of periods, are removed. These characters are uninformative and are likely to be disregarded by the models anyway.

Additionally, all text is converted to lowercase, as uppercase letters do not provide additional word information. This allows for the utilization of uncased models, which are typically smaller in size compared to their case-sensitive counterparts. These models operate with a reduced vocabulary that does not require duplicating words to consider case variations.

The dataset is structured as follows: each data instance consists of a document paired with corresponding indexes $< document, indexes >$. The $indexes$ represent the location of specific labels within the document, and take the form of $< label, index >$ pair. The $index$ comprises a starting and ending character position indicating the span of the label within the document. The $label$ is chosen from a set of six predefined sensitive labels, as outlined in section 3.1.

For each type of model described in the following sections, additional processing steps are conducted to transform the data into a format compatible with the specific structure of the model.

3.3.5 Dataset composition

Table 3.4 presents an overview of the dataset generated using BingAI. For each category of sensitive data, the document titles are listed, with the number of instances of each document, and the average character count of each document category. The dataset counts a total of 863 documents.

The same table is presented for the OpenLLaMa generated dataset. Table 3.5 has the same structure. However, it encompasses a larger quantity of documents, totaling 2051 in number. The increased document count is attributed to the presence of lower quality and accuracy in the labeling process, leading to a higher demand for additional documents to compensate for these limitations.

The presence of various types of sensitive data within the corpus text is demonstrated with Table 3.2 and Table 3.3. These tables showcase the proportion of each sensitive token relative to the entirety of the corpus. The non-sensitive category emerges as the significantly predominant one. This category comprises both fully non-sensitive documents as well as non-sensitive segments within documents that contain sensitive information.

Table 3.2: The presence of each type of sensitive data on the BingAI generated dataset.

Data type	Presence
Health	9.48%
Sexuality	1.11%
Politic	3.92%
Judicial	3.83%
Philosophical	3.58%
Ethnic	2.60%
Non_sensitive	75.47%

Table 3.3: The presence of each type of sensitive data on the OpenLLaMa generated dataset.

Data type	Presence
Health	5.87%
Sexuality	6.31%
Politic	3.51%
Judicial	6.37%
Philosophical	2.59%
Ethnic	2.04%
Non_sensitive	73.32%

Table 3.4: Description of the dataset generated with BingAI.

Data category	Doc title	Doc count	Avg char count
Mix (Sensitive)	Email	100	1463.69
	Curriculum Vitae	36	2782.78
	Articolo di giornale	30	2220
	TOT	**166**	
Health & Sexuality	Ricetta medica	25	882.44
	Referto psichiatrico	22	2852.91
	Referto psicologico	19	2945.47
	Certificato di matrimonio	23	698.65
	Certificato d’invalidità	16	1204.44
	Cartella clinica	15	2588.6
	Referto visita oculistica	12	1348.42
	Referto biopsia	9	1341.78
	Certificato di unione civile	9	983.11
	Analisi del sangue	8	1784.38
	Resoconto operazione chirurgica	8	2284.62
	TOT	**166**	
Judicial	Decreto penale di condanna	10	1729.9
	Identikit di polizia	9	1540.22
	Avviso di conclusione delle indagini	5	1642.4
	Sentenza penale	5	2750
	Certificato penale	5	1139.4
	Verbale di arresto	4	2569.5
	Partecipazione a gara d’appalto da parte di azienda	8	2226
	Certificato penale del casellario	8	1190.62
	Certificato dei carichi pendenti	7	1135.29
	Verbale di denuncia	4	1867
	Informazione di garanzia	4	1185.5
	TOT	**69**	
Politic	Endorsement politico	42	1453
	Tessera di partito	22	895.14
	Tessera sindacale	15	1501.4
	TOT	**79**	
Philosophical	Endorsement filosofico	46	1876.2
	Certificato di battesimo	4	518.75
	Attestato di partecipazione a gruppo religioso	6	802.33
	TOT	**56**	
Ethnic	Certificato di nascita	13	669.92
	Analisi del DNA	13	2485.77
	Analisi della discendenza	12	2157.25
	Analisi dell’albero genealogico	14	2560.33
	TOT	**52**	
Other (Non sensitive)	Articolo scientifico	69	3098.7
	Volantino pubblicitario per evento	63	1212.29
	Resoconto pubblicazioni scientifiche	58	2705.02
	Bilancio aziendale	11	2549
	Dichiarazione fiscale	9	2211.33
	Lettera commerciale di azienda	7	1686.57
	Contratto tra parti	6	3107
	Ordinanza di applicazione di misure cautelari	6	2732.5
	Volantino pubblicitario	5	1034.2
	Recensione di un libro	5	1906.6
	Report finanziario di azienda	5	2946
	Circolare interna	5	1567.2
	Ricetta di cucina	4	1769.25
	Fattura di azienda	4	1099.5
	Guida turistica di una città	3	2950.67
	Articolo di scienza	3	3036
	Rapporto commerciale di azienda	3	2978.67
	Catalogo di prodotti o servizi	2	2954
	Ordinanza	1	2788
	Estratto di wikipedia	50	2433.3
	TOT	**319**	

Table 3.5: Description of the dataset generated with OpenLLaMa.

Data category	**Doc title**	**Doc count**	**Avg char count**
Mix (Sensitive)	Email	233	1072.91
	Newspaper article	148	1004.84
	Curriculum Vitae	109	1241.66
	TOT	**490**	
Health & Sexuality	Psychiatric report	83	1638.61
	Medical prescription	71	1372.89
	Medical records	62	1476.06
	Psychological evaluation	53	1899.11
	Certification of invalidity	20	1419.95
	Biopsy results	21	1123.29
	Eye test report	15	1455.33
	Surgery report	15	1475.53
	Blood tests	10	1437.5
	Certificate of civil union	20	1116.95
	TOT	**370**	
Judicial	Denunciation report	26	1225.58
	Police identikit	25	1097.16
	Criminal record	24	1305.08
	Arrest report	19	1234.16
	Notice of investigation	24	1806.88
	Criminal judgement	22	1403.82
	Notice of conclusion of preliminary investigations	19	1610.89
	Certificate of pending charges	14	1268.14
	Precautionary measures	18	1281.28
	TOT	**191**	
Politic	Political endorsement	38	735.39
	Union card	30	1066.77
	Party card	28	1067.14
	TOT	**96**	
Philosophical	phylosophical endorsement	68	1028.49
	Baptismal certificate	32	783.47
	Certificate of participation to religious group	32	682.59
	TOT	132	
Ethnic	DNA analysis report	37	1804.86
	Ancestry analysis report	31	1824.16
	Birth certificate	36	435.61
	Genealogical tree report	30	1444.67
	TOT	**134**	
Other (Non sensitive)	Scientific paper	106	3038.55
	Advertising flyer for event	92	992.42
	Scientific publications report	88	2576.69
	Marriage certificate	66	829.39
	Advertising flyer	18	1035.83
	Company invoice	17	1203.71
	Services and products catalogue	15	2122.2
	Financial report by corporate	14	2215.5
	Commercial report	13	2128.92
	City travel guide	12	1598.75
	Tax declaration	12	1476.58
	Cooking recipe	11	1168.64
	Corporate memo	9	1277.67
	Company balance sheet	8	1443.12
	Book review	7	767
	Wikipedia extract	150	2430.3
	TOT	**638**	

3.3.6 Test set

The test set was generated using the BingAI chat interface. By utilizing this platform, we aimed to produce high-quality documents that closely resemble real-world scenarios, thus ensuring the authenticity and relevance of the test set.

The test set consists of 50 documents that encompass various sensitive data categories. Not all documents in the test set contain sensitive data: this deliberate inclusion of non-sensitive documents allows us to evaluate the models' ability to avoid false positives and refrain from raising unnecessary alerts.

In order to comprehensively assess the models' performance, we incorporated several "corner case" examples. These cases encompass challenging scenarios, as described in Section 3.1, and serve as a means to evaluate the models' capacity to effectively discriminate subtle differences that may be encountered in real-life use cases.

Additionally, we included document types that were not present in the training set. This approach enables us to test the models not only against new documents but also assess their ability to generalize to different document types. By exposing the models to previously unseen document types, we gain insights into their adaptability and generalization capabilities beyond the specific subset of document types used in the training phase.

Following is a table the summarizes, for each cateogry of sensitive data, the documents which are present in the test set and contains those type of data.

Table 3.6: An overview of the document types found within the test set. The classification of these documents is based on the presence of sensitive data, consideration of whether they contain such data, if they are corner cases, and their absence from the training set.

Category	Documents with sensitive data	Corner-case documents	Documents non in Test Set
Health	Email Cartella clinica Referto psicologico Esame psichiatrico Idoneità donazione sangue Esito biopsia Ricovero in pronto soccorso Certificato SERT	Estratto da wikipedia Partecipazione corso primo soccorso Pubblicazioni di primario Abstract articolo scientifico Esito ispezione salute e sicurezza Esito ispezione sanitaria Ordine nuova macchina per TAC	Partecipazione corso primo soccorso Idoneità donazione sangue Esito ispezione salute e sicurezza Certificato percorso SERT Esito ispezione sanitaria Ordine nuova macchina per TAC
Politic	Candidatura politica Curriculum Vitae Articolo di giornale Email		Candidatura politica
Sexuality	Email Curriculum Vitae Articolo di giornale Certificato transizione di genere	Seminario LGBT Abstract articolo scientifico Pubblicità su identità di genere	Pubblicità su identità di genere
Judicial	Certificato carichi pendenti Sentenza di condanna esame psichiatrico Verbale di arresto Esperienza difesa di avvocato Denuncia per stalking Identikit Difesa da parte di avvocato	Estratto da wikipedia Esperienza difesa di avvocato Estratto costituzione Curriculum Vitae	Estratto costituzione Difesa da parte di avvocato
Philosophy	Esposizione filosofica		Esposizione filosofica
Religion	Comunicato stampa ordinazione Articolo di giornale		Comunicato stampa ordinazione
Ethnicity	Analisi discendenza etnica	Estratto da wikipedia	
Other	Bilancio aziendale Email Report attività di lavoro azienda Business plan aziendale Lettera di licenziamento		Lettera di licenziamento Business plan aziendale

3.4 Models

This section outlines the theoretical foundation behind the chosen models for solving the given task, along with their respective implementations. As discussed in section 2.3, the predominant approach revolves around employing a BERT-based model. In order to explore diverse formulations of the task, we have instantiated the BERT model in three distinct architectural variations.

Considering our primary use case of performing inference in Italian, we opted for a multilingual BERT model rather than an Italian-specific BERT model. The reason for this choice was that the Italian model did not exhibit significantly better results. The multilingual model demonstrated some ability in languages beyond its specific training, but the limited testing in this regard does not warrant highlighting it, as it falls outside the primary focus of this study.

Our experimentation also includes a Spacy model as an alternative approach. The utilization of the Spacy model is particularly intriguing due to its lightweight nature and distinct architectural characteristics.

Finally, we conducted an experiment utilizing a LLM to evaluate its performance as a Zero-shot classifier. For this purpose, we employed BingAI.

3.4.1 Sentence Classification (SC)

One approach is to classify each sentence within the document being analyzed into one of six sensitive categories or a non-sensitive category. This corresponds to a multiclass text classification task, where each sentence serves as a distinct data point in the dataset.

To address this task, a classification layer is added on top of the BERT architecture. This layer takes the pooled output of the model and applies a softmax function to compute probabilities for the different classes.

The softmax function is a mathematical function that converts a vector of real numbers into a vector of probabilities that sum to one. It is often used in the output layer of a neural network to perform multi-class classification, where each output represents the probability of belonging to a certain class. The softmax function is defined as follows:

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

where z is the input vector of length K, and i is the index of the output element.

The softmax function assigns a non-zero probability to every element, which can be useful for learning and inference.

This implementation works best when all the classes involved in the classification task has an similar number of items. In case of high class imbalance, the classifier may be biased towards the majority class and ignore the minority class, resulting in poor performance. This is exacly our case: as shown in Section 3.3, the non sensitive class is in greater majority with respect to the sensitive classes. Also, for the nature of our task, we give more importance to capture as many sensitive examples as possibile, since it is preferred to be alerted of a possible sensitive data disclosure at the cost of getting some false positives.

One way to assign different weights to classes in order to counterbalance class imbalance is to use a *weighted softmax* function, which is a variant of the softmax function that incorporates a weight vector w into the formula:

$$\text{w\_softmax}(z)_i = \frac{w_i e^{z_i}}{\sum_{j=1}^{K} w_j e^{z_j}}$$

The weight vector w is chosen to reflect the inverse frequency of each class. The weighted softmax function can help to reduce the bias towards the majority class and improve the performance of the classifier on the minority class [32].

Dataset

The dataset generated in the previous section needs to be adapted to conform to the task structure at hand. If the data are in the form $< document, indexes >$, where $indexes$ is in the form $< label, index >$, it must be restructured into $< sentence, label >$. Here, each $sentence$ is obtained by tokenizing and splitting the documents into sentences, and the $label$ corresponds to one of the six sensitive classes or the non-sensitive class (a total of seven classes). A sentence is considered sensitive if at least one of its tokens is included in the sensitive $indexes$ of the original dataset.

It should be noted that the process of tokenizing documents into sentences is non-trivial. A straightforward approach is to split the document whenever a period is encountered, as it typically signifies the end of a sentence. However, there are cases where a period serves as an abbreviation for certain words, and in such cases, the sentence should not be split. To address this issue, a dictionary of frequently abbreviations has been constructed and utilized to determine whether a period represents an abbreviation or the end of a sentence. If the period follows a known abbreviation, the splitting is not performed.

Additionally, newlines can indicate the end of a sentence, especially in documents that adhere to a specific structure where a period may not always be placed at the end of sentences. Consequently, the splitting process is also performed when a newline character is encountered.

Training

The model has been trained for 3 epochs, with the following hyperparameters: batch size=16, Adam optimizer with learning rate=2e-5, weight decay=0.01. The weights for the weighted softmax loss as set as 1 for the non sensitive class, 5 for each sensitive class.

Inference

The inference pipeline for a document entails the following sequential steps:

1. Take the document as input and tokenize it into sentences.

2. Feed each sentence individually into the model.

3. Collect the output from each sentence and reconstruct the original text while maintaining a record of the results.

This inference pipeline enables the model to process the document at the sentence level, producing outputs for each sentence, and subsequently aggregating the results to obtain a comprehensive analysis of the entire document.

Although the aforementioned approach is straightforward and effective, it is prone to a fundamental limitation: even if the sentence tokenization is accurate, the resulting sentences may be too short to provide meaningful insights into their class affiliation.

Furthermore, and this serves as the primary motivation for transitioning to the subsequent approaches, as discussed in Section 3.1, determining whether a sentence is sensitive or not is also contingent upon the context in which the sentence is embedded. In the approach described earlier, the evaluation of each sentence occurs in isolation, without taking into account the contextual information provided by the surrounding text.

3.4.2 Sentence Classification with Context (SCC)

To address the limitations of the Sentence Classification (SC) model, a modified approach called Sentence Classification with Context (SCC) is proposed. The key idea is to include contextual information from the surrounding text along with each sentence to enhance the classification task.

The underlying BERT architecture remains unchanged from the previous SC model, as its suitability for the task at hand remains applicable. However, the structure of each training example is modified to incorporate the desired context.

Dataset

To embed the context of each sentence into the training examples, a method was devised to differentiate between the sentence to be classified and its corresponding context. Since BERT expects a single paragraph of text as input in the text classification task, the context itself needed to be included in the training examples.

To accomplish this, the special separator token [SEP], which is used in the training procedure of BERT to separate two sentences, was leveraged. Details regarding the usage of [SEP] in training has been discussed in Section 2.3.4.

Each training example consists of a section of text (called *chunk*), which contains a portion of the document to be classified, followed by the separator token [SEP], and the *context* in which the target chunk is located within the document. Notably, the chunks are of fixed length, thereby obviating the need for sentence tokenization. Similarly, the context also adheres to a predetermined length, ensuring consistency across the training examples.

To generate the training examples for the SCC model, a sliding window approach is employed. This involves traversing the document and creating chunks of text along with their corresponding contexts. The stride parameter determines the difference in the number of words between each example and its subsequent counterpart. By using a stride, the training examples are partially overlapping, effectively introducing a form of data augmentation.

Therefore, the structure of the dataset for the SCC model is represented as $< chunk\_context, label >$. The *chunk_context* is formed by concatenating each chunk with its corresponding context, as described earlier, while the *label* is assigned based on the class to which the chunk belongs according to the original dataset indexes.

Let's say we want to classify this portion:

`Ciao Anna,`
`come va? Ti scrivo per sapere se` **`ti sei ripresa dal raffreddore che ti ha colpito la settimana scorsa.`**

`Il raffreddore è una malattia molto contagiosa.`

Table 3.7 shows the process of splitting text into chunks using specific parameters. In this example, the chunk width is set to 7 words, the context width is set to 15 words, and the stride is set to 6 words. The purpose of this splitting approach is to include a surrounding context of words before and after each chunk.

To illustrate, let's consider the calculation for the number of words before and after each chunk. The formula used is:

$$\frac{context\_width - chunk\_width}{2}$$

For this specific case, the calculation would be $\frac{15-7}{2} = 4$

To ensure a consistent context width, it is necessary to include four words preceding and four words following each chunk. However, there are cases where the required number of words is not available, such as when a chunk appears at the beginning or end of a document. In such situations, additional words that follow or precede the chunk are appended at the end or beginning, respectively, to ensure that the context maintains the desired number of words.

Each row within the table serves as an individual training example. To illustrate this, let us consider the first row as an exemplary case. The ultimate input example presented to the BERT model can be represented as follows:

```
ciao anna come va ti scrivo per [SEP] ciao anna come va ti scrivo per
sapere se ti sei ripresa dal raffreddore che
```

The choice of parameters, namely the *context_width*, *chunk_width* and *stride*, can have varying effects on the text processing and training of a model. A longer context width can be advantageous as it allows for the inclusion of more information, enabling the model to capture a broader context. However, an excessively long context may introduce complexity and potentially confuse the model's understanding.

Also, using shorter strides can enhance the model's robustness by providing it with multiple views of similar sentences or phrases. This increased exposure to similar content can aid in learning patterns and improving generalization. However, shorter strides significantly increase the number of training examples, impacting computational resources and training time.

Table 3.7: The process of splitting a document into training example including the context. One notable observation from the lower rows of the table is that the phrase *il raffreddore è una malattia molto contagiosa*, when taken by itself, may not appear to be particularly sensitive. However, when considered within its surrounding context of *raffreddore che ti ha colpito la settimana scorsa*, it becomes evident that the reference to the cold is actually directed towards a person named Anna.

Chunk	**Context**	**Label**
ciao anna come va ti scrivo per	*ciao anna come va ti scrivo per* sapere se ti sei ripresa dal raffreddore che	NON_SENSITIVE
per sapere se ti sei ripresa dal	come va ti scrivo *per sapere se ti sei ripresa dal* raffreddore che ti ha	HEALTH
dal raffreddore che ti ha colpito la	se ti sei ripresa *dal raffreddore che ti ha colpito la* settimana scorsa. il raffreddore	HEALTH
la settimana scorsa il raffreddore è una	raffreddore che ti ha colpito *la settimana scorsa. il raffreddore è una* malattia molto contagiosa	HEALTH
una malattia molto contagiosa	raffreddore che ti ha colpito la settimana scorsa. il raffreddore è *una malattia molto contagiosa*	HEALTH

Similarly, employing shorter chunks allows for a more fine-grained scanning of the text, but at the cost of quickly increasing the number of training examples.

Training

The model has been trained for 2 epochs, with the following hyperparameters: batch size=16, Adam optimizer with learning rate=2e-5, weight decay=0.01. The weights for the weighted softmax loss as set as 1 for the non sensitive class, 5 for each sensitive class.

Inference

The inference pipeline for a document entails the following sequential steps:

1. Partition the input document into chunks and contexts, employing similar parameters such as $context\_width$, $chunk\_width$, and $stride$ as utilized during the training set division.
2. Feed each example individually into the model.
3. Collect the output from each chunk and reconstruct the original text while maintaining a record of the results.

Although this approach resolves the issue encountered in the SC model by incorporating contextual information within each chunk, it suffers from a notable drawback in terms of computational efficiency, both in the training and inference fases, when the $stride$ and $chunk\_width$ are set to lower values. In such cases, the model is required to perform inference on a significantly larger number of inputs. As a result, this limitation has prompted us to adopt an alternative strategy, wherein the problem is treated as a token classification problem instead of a sequence classification problem.

3.4.3 NER

The transition from a text classification problem to a token classification problem is a fundamental aspect of NER. As mentioned in section 2.3, NER involves the identification and categorization of significant information, known as named entities, within a given text. These named entities encompass various elements such as names, locations, companies, events, products, themes, topics, dates, monetary values, and percentages.

The essence of token classification lies in the ability to assign labels to individual tokens. By classifying each token and identifying consecutive tokens with the same label, we can concatenate them to form spans that represent specific categories.

The BERT architecture incorporates a linear layer that performs classification for each token, utilizing a softmax function to determine the most probable label for each token.

Dataset

To accomplish this, we adopt a specialized format known as BIO tagging. The BIO format, short for "beginning, inside, outside" is a widely used tagging format in chunking tasks within computational linguistics. In this format, tokens are tagged as either B (beginning), I (inside), or O (outside) of a chunk. The I- prefix denotes that the token is inside a span, while the O tag indicates that a token does not belong to any chunk. The B- prefix signifies the beginning of a chunk immediately following another chunk without any O tags in between [33]. However, in our implementation, we decided not to utilize the B tag, as we found it to have limited benefit for the model's performance. This decision was based on the fact that the frequency of chunk beginnings is relatively low compared to tokens inside and outside of chunks.

To enhance performance, we employed a technique of splitting the document into fixed-length chunks with a specified stride. This approach not only augments the data but also allows the model to focus on shorter paragraphs within the text, as opposed to processing the entire document. This strategy has proven effective in improving overall performance.

The final dataset will be in the shape $< document, indexes >$ where *indexes* is a map that assigns for each token its respective label in the format "I-" followed by one of the six sensitive categories, or "O" for the non sensitive one.

Taking as example the one proposed in the section 3.4.2, the training example would look like this:

`Ti scrivo per sapere se` **`ti sei ripresa dal raffreddore.`**

Table 3.8: This is an example of how a sentence is tokenized by the BERT tokenizer, and assigned a tag for each token.

`ti`	`scrivo`	`per`	`sapere`	`se`	`ti`	`sei`	`ripresa`	`dal`	`raff`	`##red`	`##dore`	`.`
0	0	0	0	0	1	1	1	1	1	-100	-100	-100

This table 3.8 provides an illustration of how a sentence is tokenized by the BERT tokenizer and assigned a corresponding tag for each token. Each token is associated with an integer number representing a specific label. The label mapping includes 0 (O), indicating tokens outside of named entities, 1 (I-HEALTH) representing tokens inside the named entity category "HEALTH," and -100 denoting special characters.

Notably, the BERT tokenizer employs subword tokenization (WordPiece), which allows for a hybrid approach between word-based and character-based tokenization. This technique is advantageous in handling various scenarios such as misspelled words, rare terms, and multilingual data. Subword tokenization splits certain words into subwords to enhance the model's ability to handle different linguistic elements effectively [18].

Furthermore, special tokens such as punctuation marks and end-of-sequence tokens are assigned a special tag with a significantly low value (-100). This assignment ensures that the loss function disregards these tokens during the training process. Additionally, subwords that are not the first part of a word are also assigned this value, in order to avoid double-counting the same word when computing the loss function. This way, the loss function will only consider the prediction for the first subtoken and ignore the rest. This is consistent with the way the model was pre-trained on masked language modeling, where only the first subtoken of a masked word was predicted [18].

Training

The model has been trained for 3 epochs, with the following hyperparameters: batch size=16, Adam optimizer with learning rate=2e-5, weight decay=0.01. The weights for the weighted softmax loss as set as 1 for the non sensitive class, 5 for each sensitive class.

Inference

The inference pipeline for NER on a document follows the subsequent sequential steps:

1. The input document is partitioned into fixed-length chunks using a predefined stride, employing parameters similar to those used during the division of the training set.
2. Each chunk is individually fed into the model for processing.
3. For each chunk, the tokens and their corresponding output labels are collected. Tokens with the same label that are within a distance of $token\_gap$ words, where $token\_gap$ represents a predefined parameter, are merged into a single span. This merging of smaller spans is performed because, as a token classifier, certain tokens may be left unaccounted for, resulting in small gaps within a larger span. Since our objective is to identify spans, which extends beyond the scope of token classification, the decision was made to merge adjacent smaller spans if they are sufficiently close.

3.4.4 Spacy Spancat

Spacy offers a compelling framework for various natural language processing (NLP) tasks. Specifically, we are interested in the functionality provided by Spancat (short for Span Categorization), which appears to be well-suited for our task due to its ability to handle arbitrary and overlapping spans in text data.

From a high-level perspective, Spancat can be divided into two primary components: the suggester and the classifier. The suggester is a customized function responsible for extracting potential span candidates from the input text and forwarding them to the classifier. These suggester functions can be rule-based, rely on annotations from other components, or leverage machine learning approaches.

The suggested spans are subsequently passed to the classifier, which then predicts the appropriate label for each span. By considering the contextual information encompassing the entire span, the classifier can capture informative words that reside deep within it.

Architecture

The model functions as a pipeline comprising two components: `tok2vec` and `spancat`. The tok2vec component is responsible for generating token embeddings, which are numeric representations of the tokens present in the text. The spancat component, on the other hand, is responsible for assigning labels to spans of text, such as entities or phrases.

The core architecture revolves around a CNN with a softmax layer on top, enabling computation of label probabilities for each span. The architecture is built upon pre-trained embeddings obtained from an Italian corpus, as provided by Spacy.

Training

The dataset used does not require significant modifications from its original format. The model receives input in the form of $< document, indexes >$, where *indexes* takes the form $< label, index >$.

The inference pipeline closely resembles the one presented for the NER model; thus, we will not reiterate it here.

3.4.5 BingAI

We aim to compare our results with those obtained from a LLM. LLMs have demonstrated emergent capabilities, such as Zero-Shot learning, which make them particularly interesting for our task. Given that we used an LLM to generate our dataset, it is reasonable to expect that LLMs would also perform well when tasked with performing inference.

For our comparison, we selected BingAI as our LLM of choice. BingAI is based on the technology of OpenAI GPT-4: utilizing BingAI provides us with the opportunity to leverage the cutting-edge technology offered by LLMs, while also benefiting from its availability as a freely accessible resource.

This model does not involve a training phase. Instead, it is employed for inference by providing a prompt that describes the task and the complete document. The model utilizes this input to generate an output that effectively addresses the task of identifying sensitive spans.

Prompt

The following is the prompt that is fed to the BingAI chat interface. Note that the full document needs to be attached at the end of the prompt:

```
Nel seguente testo (di finzione) possono essere contenute delle informazioni
sensibili. Una frase o porzione di frase è sensibile se riguarda informazioni
inerenti:
- alla salute di una persona (fisica o mentale) [HEALTH]
- all'orientamento politico o sindacale di una persona [POLITIC]
- all'orientamento sessuale o identità di genere di una persona [SEXUALITY]
- a procedimenti giudiziari o accuse di una persona [JURIDICAL]
- alle ideologie religiose o filosofiche di una persona [PHILOSOPHY]
- all'etnia o discendenza di una persona [ETHNIC].

Attenzione che vi sono delle frasi o porzioni di frase che rientrano
nelle tematiche di salute, politica, sessualità, dati giudiziari,
filosofici/religiosi, etnici, ma che non sono sensibili perchè non sono
riferite ad una persona. Queste frasi non devi censurarle. Fornisci il
documento con sole frasi che non hanno niente a che fare con queste
informazioni sensibili e censura le altre sostituendole con le apposite
etichette. Dal testo restituito non devo essere in grado di ricostruire
alcun'informazione sensibile.
```

The prompt used for inference is carefully constructed to closely resemble the prompt utilized during document generation. This similarity ensures that the LLM receives consistent information and context, facilitating a meaningful comparison between the generated dataset and the inference results.

Furthermore, we incorporate a negative case in the prompt to ensure that the LLM is aware of instances where sensitive spans should not be identified. This inclusion helps reinforce the model's attention to critical edge cases, enhancing its ability to discriminate between sensitive and non-sensitive spans effectively.

3.5 Evaluation

The task that we want to evaluate is a peculiar sub-category of the NLP task. In particular, we are interested in identifying in a text document the components that are to be considered as PII.

We do not only want to find the documents that contain these types of information, but also the words in the text that correspond to these data. In this sense, we can speak of span categorization.

We deal with the problem in two different ways. In the first one, we only distinguish between sensitive and non-sentitive spans, and thus sensitive and non-sensitive documents. In the second one we categorize the cluster with the six different classes we identified here above. Of course, a document may also not contain any sensitive information.

The first problem is a form of binary span categorization, while the second can be seen as multiclass span categorization. For the two problems, the evaluation changes.

3.5.1 Evaluation metrics overview

This section aims at describing the current approaches to perform evaluation on NER tasks. In *Segura-Bedmar et al.* [20] methodologies are established for effectively calculating precision, recall and F1 score. We state the definition of these three measures:

$$Prec = \frac{TP}{TP + FP}, \quad Rec = \frac{TP}{TP + FN}, \quad F1 = \frac{2 \times Prec \times Rec}{Prec + Rec}$$

Each of the methodologies descrived in *Segura-Bedmar et al.* can be seen as an algorithmic approach to derive these three numerical measures. These methodologies are:

- **Strict** Evaluation (Boundaries and Label Accuracy): This evaluation framework places a strong emphasis on achieving precise and accurate boundaries and labels. In order to be classified as a true positive, an entity must possess both exact boundaries and accurate labels. A false negative, or a missing entity, refers to an entity that remains undetected. On the other hand, a false positive, or a spurious entity, is one that is erroneously identified. Any predictions that are only partially correct, with varying combinations of accurate or partially accurate boundaries and incorrect labels, are deemed incorrect, resulting in false negatives.

- **Exact** Evaluation (Boundaries): The exact evaluation approach concentrates solely on attaining perfect accuracy in determining boundaries, irrespective of the assigned label. An entity with precisely identified boundaries, regardless of whether the label is correct or not, is considered correct and categorized as a true positive. A spurious entity is one that is present in the ground truth data but remains undetected by the model, resulting in a false positive. Conversely, a missing entity refers to an entity present in the ground truth but not detected by the model, yielding a false negative. Instances where the model incorrectly predicts boundaries, or both boundaries and labels, are classified as incorrect, leading to false positives.

- **Partial** Evaluation (Boundaries): The partial evaluation approach focuses on assessing partial accuracy in determining boundaries, while disregarding the assigned label. Therefore, an entity with partially correct boundaries and an accurate label is deemed correct. Additionally, an entity with precise boundaries but an incorrect label is also considered correct in this context, thereby classified as a true positive. A missing entity refers to an entity present in the ground truth but not identified by the model, resulting in a false negative. Conversely, a spurious entity represents an entity present in the model's predictions but absent from the ground truth, constituting a false positive. Entities with partially correct boundaries, regardless of the correctness of the assigned label, are classified as partial and therefore considered true positives. However, in this particular case, these entities are assigned a weight of 0.5, in contrast to the true positives in the exact evaluation class.

- **Type** Evaluation: The exact type evaluation focuses on accurately determining the assigned type or label, irrespective of the exact boundaries (as long as there is at least some degree of overlap). Hence, an entity with both exact boundaries and an accurate label, as well as an entity with partially correct boundaries and an accurate label, are considered correct and classified as true positives. A missing entity refers to an entity present in the ground truth but not detected by the model, resulting in a false negative. Conversely, a spurious entity represents an entity present in the model's predictions but absent from the ground truth, leading to a false positive. Any other cases, such as entities with correct boundaries but an incorrect label, or entities with partially correct boundaries but an incorrect label, are classified as incorrect, resulting in false positives.

3.5.2 Binary Evaluation

Document level

Let us consider a document $D \in \mathcal{D}$, where $\mathcal{D}$ is a corpus of documents. D is a sequence of tokens $t_i \in \mathcal{T}$, that is, $D = (t_1, \cdots, t_m)$. Consider the ground truth function $f : T \to \{0, 1\}$ that maps a token to 1 if it is part of a PII span, or to 0 otherwise, and the function associated to a model $\hat{f} : T \to \{0, 1\}$, that maps a token in a document to 1 if the corresponding model classifies the token as relevant and 0 otherwise.

We say that a document D is *sensitive* if it contains at least on token t such that $f(t) = 1$, or non-sensitive otherwise.

We can evaluate the performance of a model in classifying the documents as sensitive or non-sensitive. In this scenario, we can use the traditional measures precision, recall and f1.

These measures are immediately applicable to our problem if we define TP as the number of documents correctly classified as sensitive (they contain at least one sensitive span), FP as the number of documents that are wrongly classified as sensitive (they do not contain any sensitive span), and FN as the number of documents wrongly classified as non-sensitive (they contained at least one sensitive span but none was detected).

Span level

More complex is the situation if we want to evaluate the performances of the model at a finer level of granularity, i.e., at the level of spans. In this case we need to decide how to count the numbers TP, FN, FP.

Therefore, we took inspiration from the evaluation metrics used in *Segura-Bedmar et al.* [20], specifically the **type** evaluation system.

This **Type** evaluation algorithm is based on matching the correct type, i.e., the label, irrespective of the exact accuracy of the boundaries (as long as there is some overlap). Hence, an entity with both accurate boundaries and label and an entity with partially accurate boundaries and label are regarded as correct, thus true positives. An entity that exists in the gold standard but is not identified is missing, thus false negative. An entity that exists in the predictions but not in the gold standard is spurious, thus false positive. All other cases, i.e., a predicted entity with accurate boundaries but incorrect label; and an entity with partially accurate boundaries but incorrect label, are deemed as INC, thus among the false positives.

Therefore, in our context, let us define $GT = \{t \in D | f(t) = 1\}$ the set of tokens in the collection that belong to a sensitive span, and as $P = \{t \in D | \hat{f}(t) = 1\}$ the set of tokens in the collection that are classified as relevant. Therefore, we say that:

$$TP = |P \cap GT|, \quad FP = |P - GT|, \quad FN = |GT - P|$$

Thus, precision, recall and F1-score can be computed as above.

3.5.3 Multiclass Evaluation

The multiclass evaluation metric is derived from the binary evaluation metric: in this case, we need to compute Precision, Recall and F1 for each class. Then, the metrics obtained for each class are averaged to obtain a single value for each metric.

To compute the precision, recall and f1 for a single class, we need to count the number of *TP*, *FP* and *FN* for the specific class against all the others. In other words, we treat the case as a binary evaluation problem where the class under consideration is the positive one, and all the others are combined together as a single negative one.

Micro, macro and weighted averages

There are different ways to average the metrics obtained for each class in a multiclass evaluation problem. One way is to use the **macro-average**, which simply computes the arithmetic mean of the metrics for each class. This means that each class has the same weight in the final score, regardless of its size or frequency. Another way is to use the micro-average, which aggregates the counts of *TP*, *FP* and *FN* across all classes and then computes the metrics based on the total counts. This means that each instance has the same weight in the final score, regardless of its class. A third way is to use the weighted-average, which computes the metrics for each class and then takes a weighted mean based on the number of instances in each class. This means that each class has a weight proportional to its size or frequency in the final score.

3.5.4 Example

Let us consider another example of a multiclass evaluation problem with three classes: H (Health), J (Juridical) and O. H and J are the positive classes and O is the negative class. Suppose we have the following actual and predicted labels for eleven instances:

Token instance	1	2	3	4	5	6	7	8	9	10	11
Actual label	O	O	H	H	H	O	O	J	J	O	O
Predicted label	O	H	H	H	O	O	O	J	J	O	O

To calculate the precision, recall and f1 for each class, we need to count the number of TP, FP and FN for each class against all the others. For the class H, we have:

TP = 2 (instances 3 and 4) FP = 1 (instance 2) FN = 1 (instance 5)

Therefore, the precision, recall and f1 for the class H are:

Prec = $\frac{TP}{TP+FP} = \frac{2}{2+1} = \frac{2}{3}$ Rec = $\frac{TP}{TP+FN} = \frac{2}{2+1} = \frac{2}{3}$ F1 = $\frac{2\times Prec\times Rec}{Prec+Rec} = \frac{2\times\frac{2}{3}\times\frac{2}{3}}{\frac{2}{3}+\frac{2}{3}} = \frac{4}{6}$

For the class J, we have:

TP = 2 (instances 8 and 9) FP = 0 FN = 0

Therefore, the precision, recall and f1 for the class J are:

Prec = $\frac{TP}{TP+FP} = \frac{2}{2+0} = 1$ Rec = $\frac{TP}{TP+FN} = \frac{2}{2+0} = 1$ F1 = $\frac{2\times Prec\times Rec}{Prec+Rec} = \frac{2\times 1\times 1}{1+1} = 1$

We do not calculate the metrics for the class O, since it is the negative class.

To calculate the macro-average of the metrics, we simply compute the arithmetic mean of the metrics for each class. For example, the macro-average precision is:

$$MacroPrec = \frac{\sum_{i=1}^{n} Prec_i}{n} = \frac{\frac{2}{3}+1}{2} = \frac{5}{6}$$

where n is the number of classes (in this case, $n = 2$).

To calculate the micro-average of the metrics, we aggregate the counts of TP, FP and FN across all classes and then compute the metrics based on the total counts. For example, the micro-average precision is:

$$MicroPrec = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + FP_i} = \frac{(2+2)}{(2+2)+(1+0)} = \frac{4}{5}$$

To calculate the weighted-average of the metrics, we compute the metrics for each class and then take a weighted mean based on the number of instances in each class. For example, the weighted-average precision is:

$$WeightPrec = \sum_{i=1}^{n} w_i Prec_i$$

where w_i is the weight of the i-th class, which can be calculated as:

$$w_i = \frac{\sum_{j=1}^{m} y_{ij}}{\sum_{j=1}^{m} \sum_{k=1}^{n} y_{kj}}$$

where m is the number of instances (in this case, $m = 11$) and y_{ij} is a binary indicator that equals to 1 if instance j belongs to class i, and 0 otherwise.

In this example, the weights of the classes are:

$w_H = \frac{3}{5} \qquad w_J = \frac{2}{5}$

Therefore, the weighted-average precision is:

$$WeightPrec = (\frac{3}{5})(\frac{2}{3}) + (\frac{2}{5})(1) = \frac{4}{5}$$

4

Experiments

In this chapter, we provide a detailed summary of the results obtained through the assessment of different models on the test dataset. Moreover, we share additional observations and insights related to our experiments, including aspects such as dataset preparation, model training procedures, hyperparameter tuning, and the capabilities of LLMs in generating content.

4.1 Results

The experiments involving models' training and inference have been perfomed on a Google Colab instance with a Tesla T4 GPU, which provides 15 GB of VRAM. We analyze the following models' performance:

- **SC**: Sentence-Classification model.
- **SCC**: Sentence-Classification with Context model.
- **NER**: Name-Entity-Recognition model.
- **BingAI**: GPT-4 powered Bing Chat interface used as a Zero-shot classifier.

The Spacy model, due to inadequate results, is excluded from the analysis.

The evaluation revolves around three primary tasks:

- **Span-level classification**: This task involves accurately labeling a span, a section of contiguous words. We examine the models' performance in distinguishing each of the six sensitive classes from the non-sensitive class. Additionally, we aggregate all six classes into a single "sensitive" class and compare it against the non-sensitive class, de facto transforming the problem into a binary classification task.

- **Document-level classification**: This task evaluates the models' capability to classify complete documents into one of the six sensitive classes or the non-sensitive class. In this assessment, each document is assigned one or more labels based on the presence of at least one span corresponding to each sensitive class.
- **Inference time**: We measure the inference time of each model to evaluate their efficiency in making predictions.
- **Memory Footprint**: We assessed the disk size and memory consumption of these models when loaded.

The quality evaluation, referred to the first two metrics, was conducted on models trained using both the BingAI and OpenLLaMa datasets. To be as consistent as possible, for each type of model the same hyperparameters has been used to train them on the two datasets. This approach allowed us to assess the difference in quality between the two models.

4.1.1 Span-level results

By evaluating these models on the designated test set and considering their respective performance in the outlined tasks, we can gain insights into their comparative effectiveness.

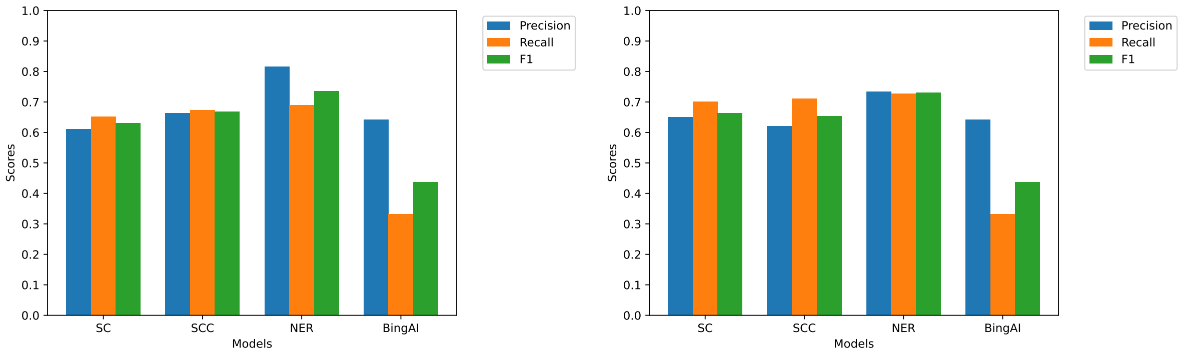
Table 4.1: BingAI dataset: Span-level metrics for each model in the multi-class setting

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Health	0.6466	0.6243	0.6352	0.7583	0.5204	0.6172	0.8051	0.8086	**0.8069**	0.8399	0.5437	0.6601
Politic	0.6946	0.6291	0.6602	0.5516	0.9817	0.7063	0.8969	0.7588	**0.8221**	0.1047	0.0245	0.0397
Sexuality	0.6754	0.3631	0.4727	0.5791	0.9001	**0.7048**	0.7791	0.3832	0.5137	0.3711	0.0528	0.0925
Juridical	0.4413	0.9017	0.5926	0.5000	0.7006	0.5835	0.8421	0.5174	**0.6410**	0.5860	0.1780	0.2731
Philosophy	0.7100	0.5233	0.6025	0.7362	0.9189	**0.8174**	0.8403	0.5886	0.6923	0.5604	0.1076	0.1805
Ethnic	0.2860	0.9391	0.4384	0.4977	0.7635	0.6026	0.5355	0.8993	**0.6713**	0.7808	0.3585	0.4914
Micro avg	0.5616	0.6518	0.6033	0.6241	0.6733	0.6477	0.8082	0.6905	**0.7447**	0.7413	0.3321	0.4584
Macro avg	0.5757	0.6635	0.6165	0.6038	0.7975	0.6873	0.7832	0.6593	**0.7159**	0.5405	0.2109	0.3034
Weight avg	0.6116	0.6518	0.6310	0.6632	0.6733	0.6682	0.8158	0.6905	**0.7355**	0.6428	0.3321	0.4379

Table 4.2: OpenLLaMa dataset: Span-level metrics for each model in the multi-class setting

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Health	0.7657	0.6789	0.7167	0.7057	0.6936	0.6996	0.8770	0.6086	**0.7185**	0.8399	0.5437	0.6601
Politic	0.6806	0.7084	**0.6942**	0.5714	0.7901	0.6632	0.5874	0.8147	0.6826	0.1047	0.0245	0.0397
Sexuality	0.4481	0.7621	0.5644	0.3897	0.7606	0.5154	0.5074	0.9985	**0.6729**	0.3711	0.0528	0.0925
Juridical	0.5035	0.6409	0.5640	0.4940	0.6916	0.5763	0.5451	0.8272	**0.6572**	0.5860	0.1780	0.2731
Philosophy	0.5727	0.8882	0.6964	0.8000	0.7004	0.7469	0.8523	0.8037	**0.8273**	0.5604	0.1076	0.1805
Ethnic	0.3282	0.8113	0.4673	0.5350	0.6729	0.5961	0.5538	0.6792	**0.6101**	0.7808	0.3585	0.4914
Micro avg	0.6126	0.7009	0.6538	0.5902	0.7105	0.6447	0.6695	0.7280	**0.6975**	0.7413	0.3321	0.4584
Macro avg	0.5499	0.7483	0.6339	0.5826	0.7182	0.6433	0.6538	0.7886	**0.7149**	0.5405	0.2109	0.3034
Weight avg	0.6497	0.7008	0.6635	0.6207	0.7104	0.6540	0.7345	0.7280	**0.7312**	0.6428	0.3321	0.4379

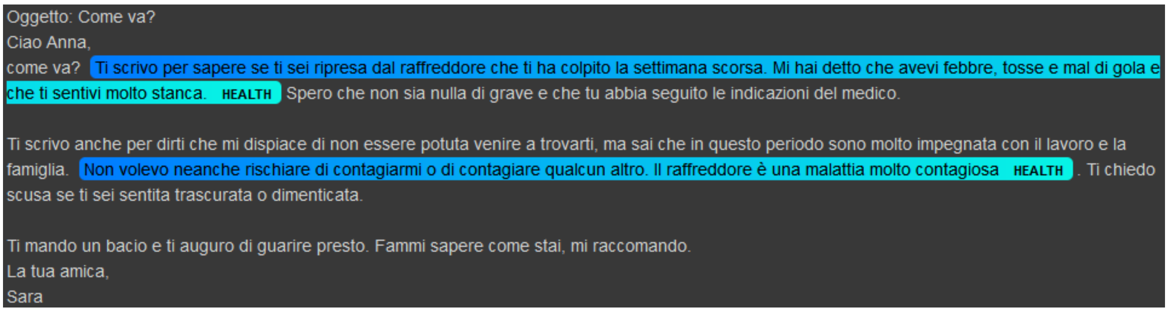
The results presented in both table 4.1 and 4.2 indicate that the NER model exhibits superior performance compared to other models in terms of F1 score, across various metrics on both the datasets. This



(a) Bing dataset

(b) OpenLLaMa dataset

Figure 4.1: Comparison of the four models on the three metrics (weighted version) in the span task.



Oggetto: Come va?
Ciao Anna,
come va? Ti scrivo per sapere se ti sei ripresa dal raffreddore che ti ha colpito la settimana scorsa. Mi hai detto che avevi febbre, tosse e mal di gola e che ti sentivi molto stanca. HEALTH Spero che non sia nulla di grave e che tu abbia seguito le indicazioni del medico.

Ti scrivo anche per dirti che mi dispiace di non essere potuta venire a trovarti, ma sai che in questo periodo sono molto impegnata con il lavoro e la famiglia. Non volevo neanche rischiare di contagiarmi o di contagiare qualcun altro. Il raffreddore è una malattia molto contagiosa HEALTH . Ti chiedo scusa se ti sei sentita trascurata o dimenticata.

Ti mando un bacio e ti auguro di guarire presto. Fammi sapere come stai, mi raccomando.
La tua amica,
Sara

Figure 4.2: Highlighted sensitive spans on a document example.

observation holds true for individual categories of sensitive data as well as the overall averages: Image 4.1 shows the result as a barplot. The slight lower quality of the OpenLlama dataset results in a few percentage points lost. Notably, the NER model outperforms the BingAI model. The reason behind this outcome can be attributed to the fact that the training dataset for the NER model incorporates documents that were not specifically generated and labeled by BingAI: this improves the generalization cababilities of the trained model. It is worth mentioning that if reference examples were provided to BingAI as part of the prompt, the results might have been considerably improved. However, in this experimental setting, our objective is to evaluate the zero-shot capabilities of the model as an out-of-the-box tool.

Figure 4.2 presents a visualization of the predicted sensitive spans on the reccurent example. The Displacy * library has been used to provide the labeled visualization.

---

*`https://spacy.io/usage/visualizers`

Binary setting

Table 4.3: BingAI dataset: Span-level metrics for each model in the binary setting.

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Average	0.6286	0.7296	0.6753	0.6960	0.7509	0.7224	0.8478	0.7243	**0.7812**	0.7480	0.3356	0.4633

Table 4.4: OpenLlama dataset: Span-level metrics for each model in the binary setting.

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Average	0.6563	0.7508	0.7004	0.6572	0.7912	0.7181	0.7257	0.7891	**0.7561**	0.7480	0.3356	0.4633

The NER model exhibits superior performance in the binary setting, as shown in both Table 4.3 and Table 4.4. This outcome aligns with our expectations, as it follows from the previous experimental setup. Notably, the evaluation metrics demonstrate an improvement in performance. By disregarding the often subtle differences between categories of sensitive data, which frequently overlap, we achieve higher levels of overall performance. This observation bears significance in light of our primary objective, which entails triggering an alert whenever sensitive data is detected in document. Irrespective of the specific category of sensitive data, the alert is triggered, underscoring the practical value of this outcome.

4.1.2 Document-level results

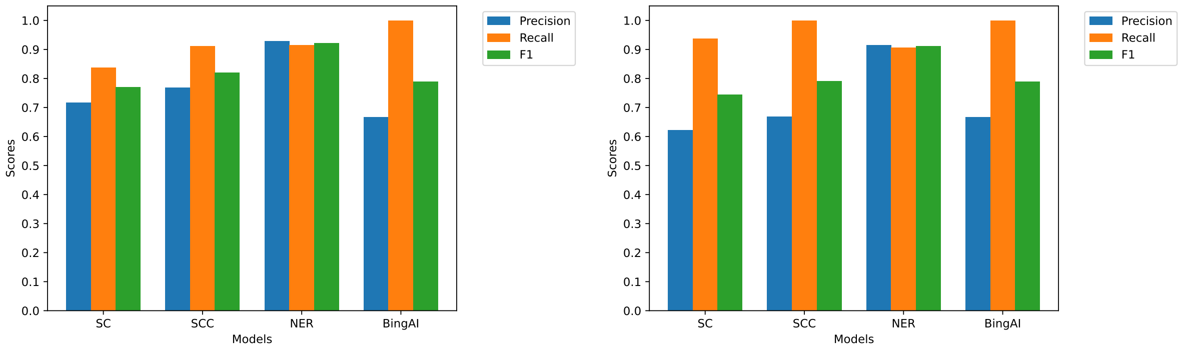
Table 4.5: BingAI dataset: Document-level metrics for each model.

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Micro avg	0.7750	1.0000	0.8732	0.7750	0.9393	0.8493	0.9142	0.9411	**0.9275**	0.7907	1.0000	0.8831
Macro avg	0.7473	0.8338	0.7851	0.7058	0.8843	0.7670	0.9345	0.8704	**0.8869**	0.5913	1.0000	0.7122
Weight avg	0.7176	0.8378	0.7704	0.7692	0.9117	0.8204	0.9296	0.9143	**0.9218**	0.6667	1.0000	0.7899

Table 4.6: OpenLlama dataset: Document-level metrics for each model.

	SC			SCC			NER			BingAI		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Micro avg	0.7142	0.9677	0.8219	0.7272	1.0000	0.8422	0.9355	0.9355	**0.9355**	0.7907	1.0000	0.8831
Macro avg	0.5619	0.8958	0.6864	0.6397	1.0000	0.7615	0.9097	0.8796	**0.8943**	0.5913	1.0000	0.7122
Weight avg	0.6216	0.9375	0.7445	0.6693	1.0000	0.7919	0.9154	0.9063	**0.9108**	0.6667	1.0000	0.7899

The document-level evaluation, as depicted in both Table 4.5 and Table 4.6 highlights the superior performance of the NER model in terms of F1. Image 4.3 illustrates this result in a barplot. Both datasets allow to train model of notable and comparable quality. This particular task holds utmost significance, as its primary aim is to detect and raise an alert when a document contains sensitive data, while also identifying its type. The evaluation metrics show a high level of efficacy, with F1 scores surpassing the 90% threshold on both micro and weighted averages.



(a) Bing dataset

(b) OpenLLaMa dataset

Figure 4.3: Comparison of the four models on the three metrics (weighted version) in the document task.

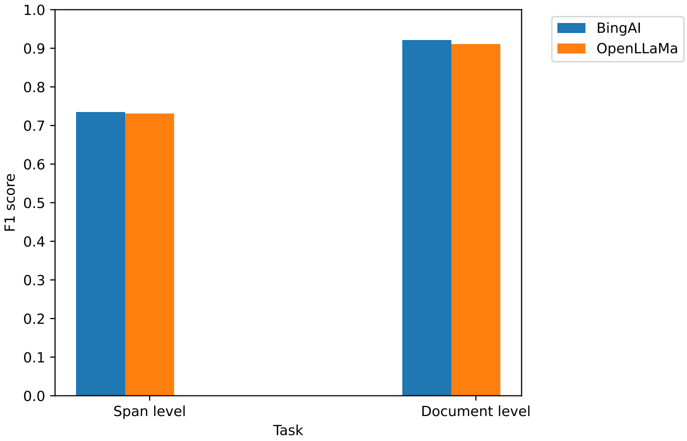


Figure 4.4: Side to side comparison between the NER models trained on BingAI and OpenLLaMa in the span and document tasks, in terms of (weighted) F1 score.

Image 4.4 shows a comparison of the NER models trained on both the datasets as a barplot graph. The results are comparable for both the span and document labeling task, with only a marginal decrease for the OpenLLaMa dataset.

4.1.3 Resource footprint

Speed of inference

Table 4.7: Performance on inference (CPU)

	SC		SCC		NER		BingAI	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
Avg time (s)	**2.0876**	**[2.0262, 2.1490]**	13.3458	[12.2521, 14.4396]	2.3033	[2.2110, 2.3956]		n.a.
Throughput (q/s)	**0.4621**	**[0.4492, 0.4750]**	0.0778	[0.0722, 0.0833]	0.4248	[0.4077, 0.4420]		n.a.

Table 4.7 presents the inference time for the three BERT models. For each metric, a 95% confidence interval is reported, obtained by collecting data from 10 distinct runs. Among them, the sentence classification model exhibits the highest speed in terms of average time per document and throughput. This is confirmed by performing a one-sided t-test between the values of the SC and the NER models, which resulted in the former being quicker than the latter with $p = 0.0007$. In fact, the NER model lags slightly behind, with a marginal difference of 0.23 seconds per document.

The evaluation presented in this study does not include the BingAI case due to various factors that influence the speed of inference. These factors, such as network connection quality, current traffic conditions, and the inability to execute the model locally, are not within our control and therefore have not been accounted for in this evaluation. Furthermore, when employing the BingAI chat interface, the process of inputting documents requires manual copy and paste, lacking automation.

On the other hand, the use of OpenAI GPT-4 introduces an additional aspect to be considered in the evaluation, namely the cost per document. As OpenAI GPT-4 is not available for free usage, the financial implications associated with its utilization need to be taken into account.

Memory usage

The BERT model utilized in our study has a disk size of 670MB. When loaded, it requires approximately 4GB of RAM for performing inference. On the other hand, the Spacy model is notable for its significantly smaller size, occupying only 40MB on disk. However, its inferior performance does not justify its usage over the highly superior BERT model.

4.2 Methodological experiments

This section compiles key findings from our experiments, which played a crucial role in shaping the methodologies discussed in Section 3. These experiments encompass areas such as dataset composition, model se-

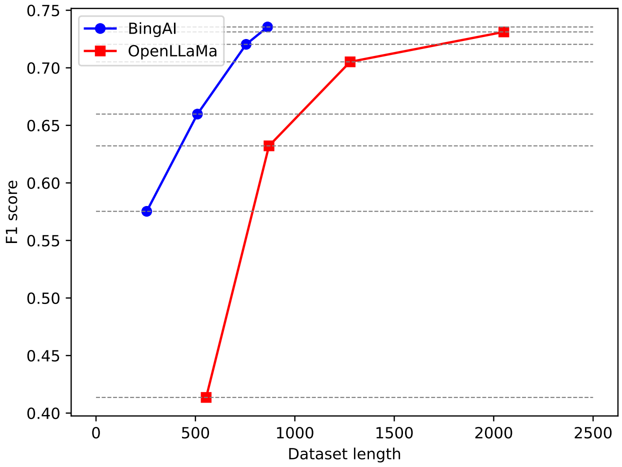


Figure 4.5: Score (weighted F1) resulting from training NER models on different size datasets, for both the BingAI and OpenLLaMa datasets.

lection, training hyperparameters, and comparisons of LLMs capabilities under various parameter settings.

4.2.1 Datasets

As outlined in Section 3.3.5, there is a notable disparity in the sizes of the training datasets generated using BingAI and OpenLLaMa. This divergence can be attributed to the presence of lower-quality and less accurate labels in the OpenLLaMa dataset, which necessitates a larger volume of documents to compensate. In Figure 4.5, we explore how the accuracy, as measured by the weighted F1 score of the trained NER model, varies with respect to the size of these two training datasets.

The BingAI dataset consistently provieds superior performance, even when composed by a limited amount of data. The BingAI dataset provides steady improvement until reaching approximately 700 documents, beyond which the score gains become marginal. Conversely, the OpenLLaMa dataset demands a significantly larger number of documents to approach similar performance levels. An F1 score of approximately 0.70 is achieved with around 1400 documents, after which the rate of improvement diminishes. We made the decision to stop data collection at slightly over 2000 documents, nearly matching the performance of the BingAI dataset, despite requiring only 863 documents for the latter.

In Section 3.3.3, we discussed the process of enriching our dataset, used for training sensitive data detection models, with external documents, including content from Wikipedia. Table 4.8 presents the results of our experiments, which explore how the model's performance changes with varying proportions of Wikipedia documents introduced into the BingAI dataset.

The table displays precision, recall, and F1-score metrics for the models at different percentages of Wikipedia documents. Specifically, it presents weighted averages for the span task and another weighted average for the document task.

The findings reveal that including some Wikipedia documents significantly enhances model performance. Both the span and document weighted averages show improvement as the percentage of Wikipedia content increases from 0% to 5%. However, introducing an excessive number of Wikipedia documents may have a detrimental impact. The span and document weighted averages decline when the percentage rises from 5% to 10%. Notably, this change is marked by an increase in precision at the expense of recall, indicating that the model becomes more cautious when identifying sensitive content, resulting in some instances where it fails to flag them. This suggests that there exists an optimal balance between the amount of Wikipedia documents and the original dataset content, and adding more Wikipedia documents does not necessarily yield superior performance. The best results are achieved when 5% of Wikipedia documents are included in the dataset.

Table 4.8: Percentage of wikipedia documents over the total document in the (BingAI) dataset

$\% \frac{\text{wikipedia}}{\text{tot\_documents}}$	0			5			10		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Span weighted avg	0.5507	0.6471	0.5884	0.8158	0.6905	**0.7355**	0.8536	0.6208	0.7188
Doc weighted avg	0.6761	0.9411	0.7762	0.9296	0.9143	**0.9218**	0.9514	0.8369	0.8904

4.2.2 Models

Section 3.4.3, explored various strategies for enhancing the capabilities of our NER model. One key aspect involved the manipulation of the dataset by adjusting the length of each document. We observed that breaking down full-length documents into smaller text chunks allowed the model to focus on more specific sections of text, leading to performance improvements. Additionally, we employed a sliding window mechanism, where a stride was used to transition from one text chunk to the next, introducing dataset augmentation.

Table 4.9 presents the results of various combinations of the length and stride parameters, measured in tokens, and how they impact performance in terms of the span labeling task and document labeling task. The table includes (weighted average) precision, recall, and F1 scores for each parameter combination.

The first case, where there's no modification to the dataset (unlimited length and 0 stride), yields relatively low performance compared to the modified parameter settings. Even limiting the document length to 500 tokens leads to notable improvements in both the span and document labeling tasks. Introducing a stride further enhances the results, and the optimal combination appears to be a document length of 100 tokens with a 20-token stride.

Table 4.9: NER model dataset formatting: performance evaluation with different combination of stride and length parameters

(length,stride)	(∞,0)			(500,0)			(500,20)			(100,20)		
	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Span weighted avg	0.8027	0.5743	0.6695	0.8222	0.5727	0.6751	0.7936	0.6854	0.7297	0.8158	0.6905	**0.7355**
Doc weighted avg	0.9313	0.7647	0.8339	0.9015	0.9017	0.9016	0.9120	0.9041	0.9080	0.9296	0.9143	**0.9218**

In Section 3.4.3, we explored an essential hyperparameter related to addressing class imbalance in the softmax function during training. Given that our datasets contained a significantly larger volume of non-

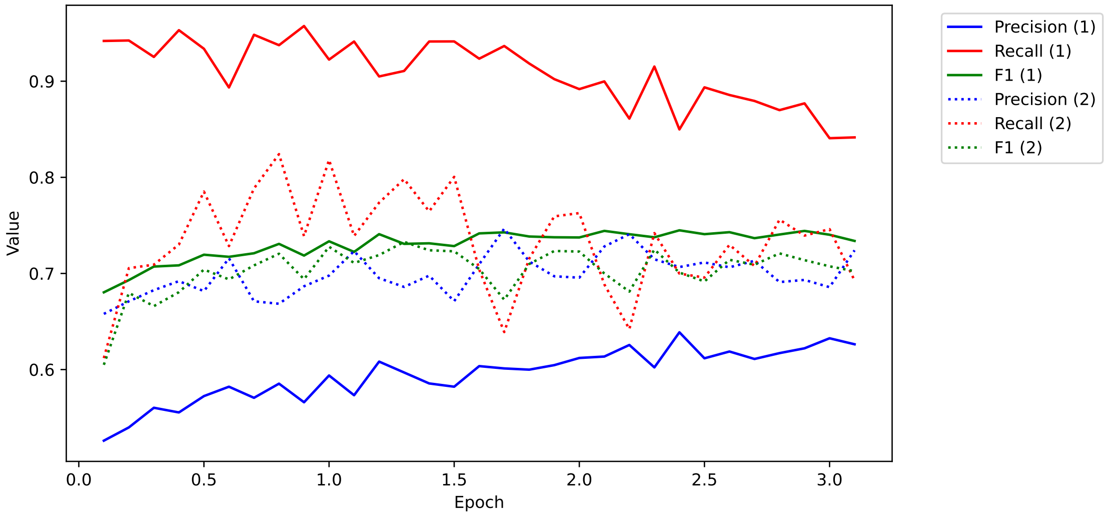


Figure 4.6: (Weighted) precision, recall and F1 for the NER model with the standard softmax loss (1 - solid lines) and for the NER model with the weighed softmax loss (2 - dotted lines

sensitive text spans compared to the six classes of sensitive data, there was a risk of introducing bias and ineffectiveness in detecting most cases of data disclosure.

Figure 4.6 illustrates the evolution of (weighted averaged) precision, recall, and F1 scores throughout the training epochs on the validation set. Dotted lines represent the NER model trained with the standard softmax weights, while solid lines represent the NER model trained with a custom weighted softmax function, assigning a weight of 5 to the sensitive class.

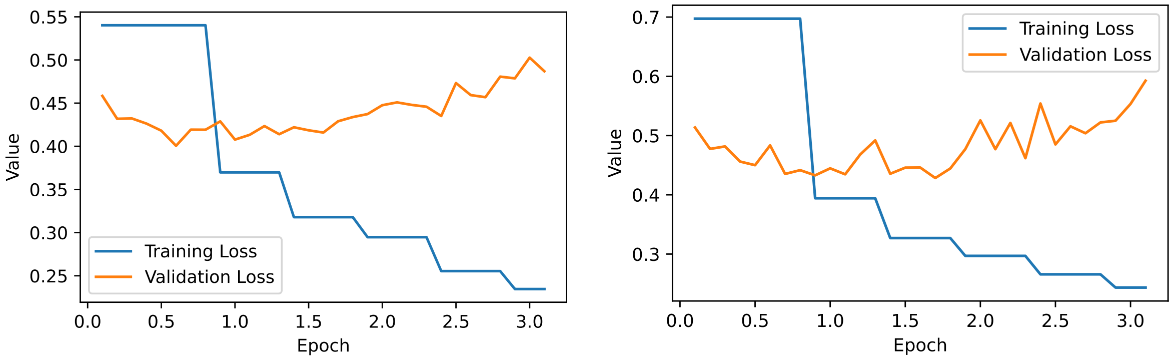
The results show that the precision, recall, and F1 scores for the standard softmax have similar values among themselves, whereas the weighted softmax results in significantly higher recall (detecting more sensitive cases) at the expense of slightly lower precision (introducing some false positives). This behavior is preferred because it's more desirable to have some false positives if it means identifying more sensitive data instances. Additionally, the resulting F1 score consistently surpasses that of the standard softmax.

The choice is further validated by the test set results presented in Table 4.10. For the weighted softmax function, both the span and document labeling tasks experience an increase of more than 10% in the (weighted) F1 score.

Table 4.10: Performance evaluation with different sensitive classes weights on the weighted-softmax loss function for the NER model

sensitive classes	weights = 1			weights = 5		
	Prec	Recall	F1	Prec	Recall	F1
Span weighted avg	0.8826	0.4558	0.6011	0.8158	0.6905	**0.7355**
Doc weighted avg	0.9375	0.6937	0.7973	0.9296	0.9143	**0.9218**

Figure 4.7 presents the progression of training and validation loss curves across training epochs for both the NER models, one employing standard softmax loss and the other using weighted softmax loss. The behavior exhibited by both models is quite similar. The training loss steadily decreases as training



(a) Weight classes = 1

(b) Weight classes = 5

Figure 4.7: Training and validation loss on the NER models when trained with the standard softmax loss (4.7a) or with the weighed one (4.7b)

progresses, but from around the second epoch onward, the validation loss begins to rise.

Notably, the results depicted in the score curves in Figure 4.6 suggest that improvements continue even when the validation loss starts increasing, indicating a relative risk of overfitting. Consequently, we opted to halt training at approximately the third epoch. This protocol for determining the training epoch was employed consistently across all our models.

4.2.3 LLMs generation parameters

In this section, we provide examples of documents generated using LLMs, specifically OpenLLaMa, with the prompts and parameters discussed in Section 3.3. Assessing the quality of these generated documents can be somewhat subjective, as there is no definitive measure. We determined the best outcomes through a combination of theoretical considerations and trial-and-error, evaluating the document quality visually. Also, LLM-generated text can be non-deterministic, meaning that even with the same configuration settings, different text can be generated (unless the temperature is set to 0). However, the key qualities and characteristics we analyze here remain consistent across multiple generations.
To illustrate this, we begin with the following prompt:

```
Please generate a fictional but realistic document referred to Olivia of the
type 'Psychiatric report', which explains in great detail health information
about Olivia.
```

We present several generated documents with different configurations.

For instance, Text 4.8 showcases the outcome of generation using the custom instruction prompt and parameters detailed in Section 3.3.1. This document is characterized by its length, coherence, and level of detail, real-case names (drugs) which align with our objective of accurately replicating real-case documents.

In contrast, Text 4.9 was generated using the standard Alpaca instruction template. While the results are coherent, they tend to be shorter and more concise, lacking variability. This is similar to the effect of reducing the temperature parameter, as demonstrated in Text 4.10. With a temperature of 0.3, the resulting document appears brief and superficial, making it less useful for our purposes.

Increasing the temperature introduces more variability, but excessive values can yield incoherent text that lacks logical sense, as observed in Text 4.11.

Lastly, we attempted to generate a document in the Italian language. As discussed in Section 3.3.1, the OpenLLaMa model may not exhibit sufficient proficiency in generating text in languages other than English. Text 4.12 appears coherent, but its choice of vocabulary appears highly unusual to a native Italian speaker, often being a literal translation of English terms without contextual adaptation. This does not align with the style of Italian documents, whereas translation services can provide more natural results.

```
Hi there,

I'm Dr. Danforth, and I'm writing this psychiatric report on Olivia for Dr.
Miller. Olivia has been struggling with depression and anxiety for several
years now, and her condition has worsened recently. She is currently taking
several medications for these symptoms, including Zoloft and Ativan. Olivia
reports that she often feels overwhelmed by negative thoughts and feelings
of hopelessness, especially during stressful periods. She also describes
having difficulty sleeping at night, which can worsen her mood during the day.
Overall, Olivia's mental health has significantly impacted her daily life and
relationships, and she is seeking additional support and treatment to improve
her well-being.

I have conducted a thorough medical examination of Olivia, and I am confident
that her symptoms are real and require attention and treatment. Olivia's
mental health issues have been a persistent problem throughout her life,
and they have had a significant impact on her overall wellbeing. During
our conversation, Olivia reported that she often feels like she is "going
crazy" or that everything is "all wrong." She also expressed worry about her
future and fear of what might happen if she does not receive proper treatment.
Overall, Olivia's self-reported symptoms are consistent with a diagnosis of
major depressive disorder, severe impairment, and a need for intensive and
comprehensive treatment.

Based on my clinical experience and examination of Olivia, I strongly recommend
that she receive intensive and comprehensive treatment for her major depressive
disorder. This will include a combination of medication and psychotherapy,
along with a focus on healthy lifestyle habits such as regular exercise and
good sleep hygiene. Additionally, I recommend that Olivia engage in regular
social interaction and seek support from friends and family members to help
manage her symptoms. By working together, we can develop a comprehensive
treatment plan that will help Olivia achieve lasting improvement in her mental
health.

Thank you for your time today, Dr. Miller. I look forward to continuing our
discussion and developing a treatment plan that will help Olivia achieve her
desired outcomes.

Sincerely, Dr. Danforth Assistant Professor of Psychiatry University Medical
Center Los Angeles, CA Date: March 14, 2023
```

Figure 4.8: Example of generation of a document with the custom instruction template

```
Olivia, 36 years old, was admitted to the psychiatric unit on Wednesday, March
21, 2023, for evaluation and treatment of major mental illness. She had been
experiencing symptoms of severe depression, anxiety, and insomnia for several
weeks prior to her admission. During her stay at the hospital, she has been
receiving treatment with various medications and psychotherapy. The staff
members have found Olivia to be cooperative and polite during her stay. Her
condition has improved significantly over the past few days, and she is no
longer considered a danger to herself or others. It is recommended that she
remain in treatment until her symptoms have subsided completely, after which
she can be discharged.
```

Figure 4.9: Example of generation of a document with the standard instruction template

```
Olivia is a 32-year-old female who has been struggling with severe depression
and anxiety for many years. She has been seeing a therapist for the past five
years and has been on various medications to help manage her symptoms. She has
had several major life stressors in the past few years, including a difficult
relationship with her parents, a traumatic incident at work, and a miscarriage.
Her mental health has deteriorated significantly over the past year, and she
has been having suicidal thoughts and has made multiple attempts to harm
herself. She is currently not taking any medication for her mental health
issues and is not in a good place emotionally. She needs to be hospitalized
immediately to get the help that she needs.
```

Figure 4.10: Example of generation of a document with the custom instructions and temperature = 0.3

```
This is Dr. Carol James of Community Mental Health Research Clinic, writing
this psychiatric report for Olivia Dunn, for use by her medical providers
and care team. While solely diagnosing and treating mental health conditions
remains our primary focus, there are certainly interrelated physical health
risks associated with severe mental illness. In order to render comprehensive
treatment and ensure whole-person healing, we at CMHRC work diligently to
monitor medical conditions as part of our integrated mental and physical
healthcare approach. Diagnoses vs trigger situations can alter without notice.
Please check back regularly for updates and changes. This report will discuss
any potentially harmful triggers, current diagnoses, vaccinations, medication
assisted treatment (MAT), polypharmacy, usual treatment modalities, campus
resources, program Admissions policy and assessment disposition.
Many stabilization methods are available through recovery based programming.
What defines length of stay is very individualized, mainly contingent on
symptom relief and social situations remain dormant with success maintained
after discharge, initially working from the Individualized Treatment Plan with
designated Phase - Sym. should have trajectory for Wellness RecoverySAMPLE
and have housing arrangement-avoid living situation PLUS best Supported living
arrangements can have services available close to home, if these are necessary
Spend their final
```

Figure 4.11: Example of generation of a document with the custom instruction template and temperature = 1.5

```
Olivia è stata ricoverata presso l'ospedale psichiatrico di Firenze per una
serie di comportamenti molto strani e poco comuni, che hanno portato alla
sospensione della sua normalità quotidiana. Le analisi cliniche hanno mostrato
alcune caratteristiche molto interessanti nella sua personalità, tra cui
mancanza di empatia, agitazione emotiva, disturbo schizoidico di personalità,
trascuranza nei confronti degli obblighi sociali, dipendenza dall'alcol e
da droghe, impulsività ed altre forme di attenzione deficit disorder. La
condotta di Olivia ha sempre suscitato preoccupazione nei familiari e amici,
tanto più che le sue atteggiamenti sono state anche violenti verso questi
ultimi. Dopo essere stata ricoverata, Olivia ha mostrato un certo grado di
cambiamento, dimostrando un buon rispondimento all'assistenza medica e alle
terapie. Per questi motivi, i medici hanno deciso di evitare ogni ulteriore
ricovero psichiatrico di Olivia, accontentandosi di monitorarla con un medico
residente e continuando a seguirla durante la cura.
Per quanto riguarda la vita privata di Olivia, deve ancora affrontare una
serie di problemi personali che potrebbero costituire un'altra minaccia per
la propria salute mentale. Tuttavia, gli ospedali psichiatrici possono essere
considerati come un punto di partenza, non una meta. Speriamo che Olivia
possa riuscire a gestire le sue problematiche mentali, diventando una persona
autosufficiente e soddisfatta nella vita.
Sottolineiamo che Olivia ha subito qualche episodio di depressione nell'ultimo
periodo, che può aver contribuito a provocare i suoi comportamenti molto
strani. Il suo caso dovrebbe essere studiato con maggiore attenzione dai
medici, per capire come tentare di evitare lo scoppio di nuovi episodi di
depressione.
```

Figure 4.12: Example of generation of a document in italian

5

Conclusion

This thesis has presented a novel approach for identifying sensitive data in text documents, based on the GDPR framework and the investigative domain. The main contributions of this work are:

- A comprehensive review of the state-of-the-art techniques for PII and sensitive data detection, highlighting the challenges and limitations of existing methods.
- A proposal of a new framework for sensitive data detection, covering six key domains and four critical criteria, emphasizing the importance of context and sensitivity spans.
- A novel method for generating synthetic datasets using LLMs, comparing two different LLMs (BingAI and OpenLLama) and demonstrating the benefits of incorporating external sources for document generation and span labeling.
- An extensive evaluation of various models for sensitive data detection, including BERT-based models with various adaptation to perform text classification and NER, CNN models.
- A definition of metrics to evaluate the performance of a NER task that involves the identification of span of text.

The results of this thesis show that sensitive data detection is a feasible and valuable task, especially in the context of data protection and privacy, with a 90% of score for the best performing BERT model. The proposed framework and method can be applied to various domains and scenarios, such as journalism, law enforcement, healthcare, and social media. The use of LLMs for synthetic data generation offers a promising solution for overcoming the scarcity of real-world data, while preserving the diversity and quality of the data: also, smaller LLMs model which can be run from local environment are able to produce text of a sufficient quality to train classification models that perform nearly as well as models trained on much higher data quality.

References

[1] "Regulation (EU) 2016/679 of the European Parliament and of the Council," *Regulation (eu)*, vol. 679, p. 2016, 2016.

[2] T. Paccosi and A. P. Aprosio, "REDIT: A tool and dataset for extraction of personal data in documents of the public administration domain," in *Proceedings of the Eighth Italian Conference on Computational Linguistics, CLiC-it 2021, Milan, Italy, January 26-28, 2022*, E. Fersini, M. Passarotti, and V. Patti, Eds., ser. CEUR Workshop Proceedings, vol. 3033, CEUR-WS.org, 2021.

[3] Microsoft, *Microsoft/presidio: Context aware, pluggable and customizable data protection and de-identification sdk for text and images*, Accessed: July 2023.

[4] M. Petrolini, S. Cagnoni, and M. Mordonini, "Automatic detection of sensitive data using transformer-based classifiers," *Future Internet*, vol. 14, no. 8, p. 228, 2022.

[5] G. Gambarelli, A. Gangemi, and R. Tripodi, "Is your model sensitive? SPEDAC: A new resource for the automatic classification of sensitive personal data," *IEEE Access*, vol. 11, pp. 10 864–10 880, 2023.

[6] F. Almeida and G. Xexéo, "Word embeddings: A survey," *arXiv preprint arXiv:1901.09069*, 2019.

[7] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, 2014, pp. 1746–1751.

[8] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132 306, 2020.

[9] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[10] J. Wei, X. Wang, D. Schuurmans, *et al.*, "Chain of thought prompting elicits reasoning in large language models," *CoRR*, vol. abs/2201.11903, 2022. arXiv: `2201.11903`.

[11] B. Clavié, A. Ciceu, F. Naylor, G. Soulié, and T. Brightwell, "Large language models in the workplace: A case study on prompt engineering for job type classification," *CoRR*, vol. abs/2303.07142, 2023. arXiv: `2303.07142`.

[12] J. White, Q. Fu, S. Hays, *et al.*, "A prompt pattern catalog to enhance prompt engineering with chatgpt," *CoRR*, vol. abs/2302.11382, 2023. arXiv: `2302.11382`.

[13] H. Mao, X. Shuai, and A. Kapadia, "Loose tweets: An analysis of privacy leaks on twitter," in *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society, WPES 2011, Chicago, IL, USA, October 17, 2011*, Y. Chen and J. Vaidya, Eds., ACM, 2011, pp. 1–12.

[14] A. G. Pablos, N. Pérez, and M. Cuadros, "Sensitive data detection and classification in spanish clinical text: Experiments with BERT," *CoRR*, vol. abs/2003.03106, 2020. arXiv: `2003.03106`.

[15] B. Klimt and Y. Yang, "The enron corpus: A new dataset for email classification research," in *Machine Learning: ECML 2004, 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004, Proceedings*, J. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, Eds., ser. Lecture Notes in Computer Science, vol. 3201, Springer, 2004, pp. 217–226.

[16] S. Wang, Y. Liu, Y. Xu, C. Zhu, and M. Zeng, "Want to reduce labeling cost? GPT-3 can help," *CoRR*, vol. abs/2108.13487, 2021. arXiv: `2108.13487`.

[17] B. Chintagunta, N. Katariya, X. Amatriain, and A. Kannan, "Medically aware GPT-3 as a data generator for medical dialogue summarization," *CoRR*, vol. abs/2110.07356, 2021. arXiv: `2110.07356`.

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

[19] X. Geng and H. Liu, *Openllama: An open reproduction of llama*, May 2023.

[20] I. Segura-Bedmar, P. Martínez, and M. Herrero-Zazo, "Semeval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013)," in *Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2013, Atlanta, Georgia, USA, June 14-15, 2013*, M. T. Diab, T. Baldwin, and M. Baroni, Eds., The Association for Computer Linguistics, 2013, pp. 341–350.

[21] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[22] F. Karl and A. Scherp, "Transformers are short text classifiers: A study of inductive short text classifiers on benchmarks and real-world datasets," *arXiv preprint arXiv:2211.16878*, 2022.

[23] G. Ateniese, G. Felici, L. V. Mancini, A. Spognardi, A. Villani, and D. Vitali, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *CoRR*, vol. abs/1306.4447, 2013. arXiv: `1306.4447`.

[24] C. Dwork, A. Smith, T. Steinke, and J. Ullman, "Exposed! a survey of attacks on private data," *Annual Review of Statistics and Its Application*, vol. 4, Mar. 2017.

[25] T. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[26] OpenAI, "Gpt-4 technical report," *ArXiv*, vol. abs/2303.08774, 2023.

[27] H. Touvron, T. Lavril, G. Izacard, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[28] C. Wang, S. X. Liu, and A. H. Awadallah, "Cost-effective hyperparameter optimization for large language model generation inference," *CoRR*, vol. abs/2303.04673, 2023. arXiv: `2303.04673`.

[29] M. U. Hadi, q. tashi qasem, R. Qureshi, *et al.*, *Large language models: A comprehensive survey of its applications, challenges, limitations, and future prospects*, Jul. 2023.

[30] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. J. Anderson, "The curse of recursion: Training on generated data makes models forget," *CoRR*, vol. abs/2305.17493, 2023. arXiv: `2305.17493`.

[31] B. Chowdhury and G. Garai, "A review on multiple sequence alignment from the perspective of genetic algorithm," *Genomics*, vol. 109, Jun. 2017.

[32] H. Zhu, Y. Yuan, G. Hu, X. Wu, and N. Robertson, "Imbalance robust softmax for deep embedding learning," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[33] L. A. Ramshaw and M. Marcus, "Text chunking using transformation-based learning," in *Third Workshop on Very Large Corpora, VLC@ACL 1995, Cambridge, Massachusetts, USA, June 30, 1995*, D. Yarowsky and K. Church, Eds., 1995.

Acknowledgments

This work is a culmination not only of my effort but also of the support and encouragement from the wonderful people in my life.

My deep appreciation goes to my supervisor, Alberto Testolin, whose consistent presence and constructive guidance have played an absolute role in refining this work.

I extend my sincere gratitude to Dennis Dosso, my company co-supervisor, for his continual support, both technically and personally, which has greatly influenced the success of this research.

A special thanks also goes to Daniele Turato and my colleagues at Siav S.p.A. for their warm welcome and supportive environment during my internship.

To my parents, Attilio and Monica, for their incredible support throughout my academic journey. Your encouragement, especially during challenging moments, has been a constant source of strength.

Additionally, I express my gratitude to all my friends for turning this academic ride into a fun and unforgettable experience. Your companionship made the challenges easier and the successes more enjoyable.

This work is my way of acknowledging each of you. It wouldn't have been the same without your incredible presence in my life.