UCLouvain

École polytechnique de Louvain

Deep Learning for de-identification of clinical documents

Author: **Nicolas HUBERTY**
Supervisor: **Sébastien JODOGNE**
Readers: **Marie-Catherine DE MARNEFFE, Alexandre ENGLEBERT**
Academic year 2023–2024
Master [120] in Computer Science

ABSTRACT

The amount of unstructured medical documents increases each year, presenting an opportunity to extract valuable insights that could significantly improve healthcare. However, to take advantage of this potential, it is crucial to de-identify these documents in order to protect patient privacy and to be able to use these documents for research. This study will explore the different deep learning solutions for the de-identification of clinical documents. The first part explores the current strategies to recognize specific words in documents to understand which method has the greatest impact on performances. This evaluation helps to identify the strengths and weaknesses that traditional deep learning approaches may encounter.

The second part will introduce an innovative open-source tool: the Incremental Learning Annotator (ILA). This tool enhances the ability to obtain quickly a robust model that achieves good performance. This solves the need of large and well annotated dataset to obtain a robust deep learning model.

Acknowledgments

I am deeply grateful to my supervisor, Sébastien Jodogne, for his support, consistent availability, dedicated involvement, and guidance throughout my research. I extend my gratitude to Amaury Fierens, whose expertise and advice have been invaluable in navigating through the complexities of this study. Their contributions have significantly enhanced the quality of my work.

I am also grateful to the Electrokot, whose support and encouragement were needed during this academically intense year.

More personally, I must express my appreciation to my family for their unwavering support and encouragement. Their belief in my abilities has been a constant source of motivation. Additionally, I owe a special thanks to my girlfriend, Charlotte Turpin, for her patience, understanding, and the countless hours she spent listening to my ideas and challenges, helping me to maintain my focus and persevere through this master thesis.

Special thanks are also due to the clusters CISM for providing the computational resources necessary for this research.

Each of you has played an essential role in this work, and for that, I am eternally grateful.

CONTENTS

Abstract **i**

Acknowledgments **ii**

1 Introduction **1**
1.1 Background and Motivation 1
1.2 Problem Statement 2
1.2.1 Single-Dataset Focus 2
1.2.2 Cross-Dataset Application 3
1.3 Research Objectives 4

2 Literature Review **5**
2.1 Summary of Dataset Availability 5
2.1.1 MedDocAn 5
2.1.2 i2b2 2006 6
2.1.3 i2b2 2014 7
2.1.4 multiNERD 8
2.1.5 wikiNER 9
2.2 Historical Review of Different Approaches 9
2.2.1 Human Annotation 10
2.2.2 Rule-Based 10
2.2.3 Machine Learning and Hybrid System 11
2.2.4 Deep Learning 12
2.2.5 Transfer Learning 13

3 Theoretical Foundations **14**
3.1 Tokens and Tokenizers 14
3.2 Transformer Models 15
3.3 *BERT* Model 16
3.4 *BERT* Justification 17
3.5 Training Parameters 17

3.6 Incremental Learning . . . 18
3.7 BIO-Tagging . . . 18
3.8 Frameworks and Tools . . . 18

4 Evaluation of Current De-Identification Methods 19
4.1 Methodology . . . 19
4.1.1 Dataset Format Used . . . 19
4.1.2 Baseline Experimental Setup . . . 20
4.1.3 Loss Function . . . 21
4.1.4 Evaluation Metrics . . . 21
4.1.5 Evaluation Methods: Entity-Based vs. Token-Level . . . 22
4.2 Evaluation of Traditional Approaches . . . 23
4.2.1 Precise Results of *RoBERTa* on i2b2 2014 . . . 24
4.2.2 Label Definition . . . 24
4.2.3 Language Performances . . . 25
4.2.4 Impact of the model . . . 27
4.2.5 Parameters Performances . . . 30
4.3 Discussion of Traditional Approaches . . . 32
4.3.1 Evaluation of BIO-Schema Removal . . . 32
4.3.2 Effectiveness of Group Labeling . . . 32
4.3.3 Language Model Performance Analysis . . . 33
4.3.4 Impact of the Model . . . 34
4.3.5 Optimization of Model Parameters . . . 35
4.3.6 Conclusion . . . 35
4.4 Evaluation of Transfer Learning . . . 36
4.4.1 Naive Transfer Learning . . . 36
4.4.2 Impact of the Fine-Tuning Process . . . 37
4.4.3 Speed of Training without Transfer Learning . . . 38
4.5 Discussion of Transfer Learning . . . 38
4.5.1 Effectiveness of Naive Transfer Learning (without Fine-Tuning) . . . 38
4.5.2 Future Directions . . . 40
4.6 Conclusion . . . 40

5 Development and Evaluation of the Incremental Learning Annotator 41
5.0.1 How the Tool Addresses the Problem Statement . . . 41
5.0.2 General Concept of the Incremental Learning Annotator . . . 42
5.1 Operational Methodology . . . 44
5.1.1 General Architecture . . . 44
5.2 Technical Workflow . . . 46
5.2.1 Creation of the Project . . . 46

5.2.2 Manager: Orchestrating Model Operations . . . 47
5.2.3 Training the Model . . . 48
5.2.4 Prediction and Alignment . . . 49
5.2.5 Evaluation of Model Performance . . . 49
5.3 Practical Information . . . 51
5.3.1 Tool Interface and Usage . . . 51
5.3.2 API Access and Integration . . . 53
5.4 Results . . . 53
5.4.1 Evaluation Strategy . . . 53
5.4.2 Document Selection Strategies . . . 54
5.4.3 Confidence of the Model . . . 55
5.4.4 Impact of Training Data Quality on Model Performance . . . 56
5.4.5 Selective Model Strategy . . . 57
5.5 Discussion . . . 57
5.5.1 Evaluation Strategy and Results . . . 57
5.5.2 Lowest Confidence First . . . 58
5.5.3 Model Confidence Over Iterations . . . 58
5.5.4 Impact Analysis of Label Omission and Model Robustness . . . 58
5.5.5 Discussion on Selective Model Improvement Strategy . . . 59

6 Conclusion **60**
6.1 Summary of Findings . . . 60
6.1.1 Challenges with Traditional and Transfer Learning Approaches . . . 60
6.1.2 Innovative Solution: Incremental Learning Annotator (ILA) . . . 60
6.1.3 Comparative Analysis and Findings . . . 61
6.2 Contributions to the Field . . . 61
6.2.1 Development of Open-Source Tools . . . 61
6.2.2 Innovative Incremental Learning Annotator (ILA) . . . 61
6.3 Limitations and Challenges . . . 61
6.3.1 Tokenization . . . 62
6.3.2 Token-Level Assessment . . . 62
6.3.3 Lack of Diversity in ILA Evaluation . . . 62
6.3.4 User Interface Limitations of the ILA . . . 62
6.4 Recommendations for Future Research . . . 62
6.4.1 Testing with Advanced Deep Learning Models . . . 63
6.4.2 Quantitative Evaluation of Tool Efficiency And Time winning . . . 63
6.4.3 Expansion of Evaluation Datasets . . . 63
6.4.4 Improving Code Modularity and Integration . . . 63
6.5 ChatGPT . . . 63

Appendices **64**

A **64**

A.1 Mappings . . . 64
A.1.1 Meddocan Mapping . . . 64
A.1.2 i2b2 2006 Mapping . . . 65
A.1.3 i2b2 2014 Mapping . . . 66
A.1.4 multiNERD Mapping . . . 67
A.2 Roberta Code . . . 67
A.3 Description of Machine Learning key techniques . . . 69
A.4 Historical Approaches of Deep Learning . . . 69
A.5 General Deep Learning Concepts . . . 70
A.5.1 Understanding Neurons: Weights, Biases, and Activation Functions . . . 71
A.5.2 Introduction to Natural Language Processing . . . 71
A.5.3 Recurrent Neural Network (RNN) . . . 72
A.5.4 Challenges: Vanishing and Exploding Gradients in RNNs . . . 72
A.5.5 Mitigation with LSTM Units . . . 73
A.5.6 From Sequential to Contextual Processing: RNNs vs. Self-Attention . . . 74
A.6 Evaluation . . . 75
A.6.1 Dataset Format . . . 75
A.6.2 Grouping Labels Results . . . 77
A.6.3 Model Performance on i2b2 2006 Dataset . . . 78
A.6.4 Model Performance on *MEDDOCAN* Dataset . . . 79
A.6.5 Batch Size Performances with i2b2 2006 . . . 79
A.6.6 Learning Rate Performances with MEDDOCAN . . . 80
A.6.7 mBERT (Multilingual BERT): . . . 80
A.6.8 sp (Spanish BERT - BETO): . . . 80
A.7 ILA . . . 81
A.7.1 Database Columns . . . 81
A.7.2 Configuration File . . . 81

Bibliography **82**

Chapter 1

Introduction

1.1 Background and Motivation

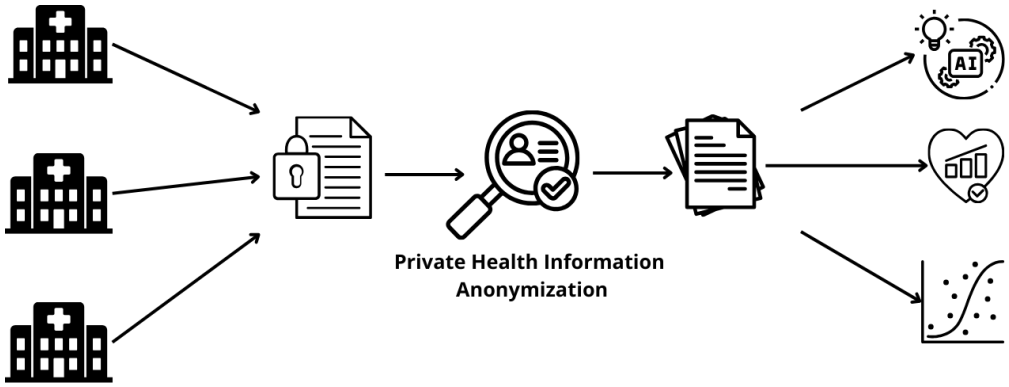


Figure 1.1: Process of Anonymizing Protected Health Information

Last years, the healthcare domain has experienced an unprecedented digital transformation thanks to widespread adoption of Electronic Health Records (EHRs) that significantly enhance healthcare efficiency, as demonstrated by several studies [4, 10]. This evolution has made it possible to use these data to improve healthcare systems, offering data-driven diagnostics, personalized treatment or predictive healthcare outcomes.

Protecting health information is vital. This ensures compliance with laws like *Health Insurance Portability and Accountability Act (HIPAA)* or *General Data Protection Regulation (GDPR)*. It also helps maintain patient trust and prevents harm like identity theft and discrimination. Medical data is sensitive. Unauthorized access, such as by insurers can be dangerous. This underscores the need for complete and accurate de-identification.

The main challenge with using these data is anonymizing all medical documents that leave the hospital for research purposes. The *HIPAA* defines certain Protected Health Information (PHI) that must be removed to protect patient identity [1.1].

De-identifying clinical data is challenging due to its complexity. Manual annotation is one method used to achieve this, but it is both time-consuming and costly, making it impractical for the large volumes of data generated daily. Therefore, more efficient and automated methods are needed to handle the growing amount of clinical data while maintaining high accuracy.

Although rule-based methods and traditional machine learning models can be used, but they often do not address all privacy and efficiency challenges in modern healthcare datasets. This issue has increased interest in Deep Learning technologies, which offer efficient and automated solutions to the de-identification problem, as demonstrated by several studies[13, 33].

However, despite the progress in Deep Learning, there is a shortage of publicly available annotated medical datasets. This creates a cycle that slows down progress. Deep Learning has the potential to revolutionize medical document de-identification, but there is a gap between its capabilities and real-world use. Without enough data, training effective models is hard. Without effective models, processing new data for research is impossible.

1.2 Problem Statement

The de-identification of clinical documents presents several challenges due to the complexity and variability of healthcare data. These challenges can be tackled using two distinct approaches: single-dataset focus and cross-dataset application.

1.2.1 Single-Dataset Focus

This approach involves developing models using a single dataset from a specific institution, such as a hospital. The training and testing sets consist of documents coming from the same institution, ensuring the model is exposed to consistent data types. Consequently, the model becomes adept at handling documents it has already encountered ("seen" documents), as it has been trained on data containing all the institution-specific information. This approach, while efficient has one main disadvantage:

- **Costs of High-Quality Annotated Datasets:** High-quality annotated documents are crucial for training de-identification models. Actually, the single *HIPAA* compliant solution that provides high quality annotations is the manual annotation. However, this is time-consuming and expensive and this is an obstacle for hospitals and other centers, which would have to pay

a considerable amount of money to have a good model created for their data, and to be able to make their documents public.

1.2.2 Cross-Dataset Application

Another approach is to leverage models trained on existing annotated datasets from one healthcare institution and apply them to clinical documents of another institution. This approach removes the task of annotating documents, as it uses documents that have already been annotated. However, this approach faces significant problems, primarily because methods generally **do not perform well on unseen data**. Several factors contribute to this issue:

- **Unstructured and Diverse Data Formats:** Clinical documents vary significantly in structure and format across different healthcare institutions. The absence of standardized formats for medical notes and clinical communications introduces more complexity.
- **Language Diversity:** Models trained on datasets in one language may not perform well on data of another language, limiting the applicability of a single model across diverse linguistic environments.
- **Sparse and Inconsistent Labeling:** Datasets are few and often have inconsistent labels. For example, the *HIPAA Safe Harbor Act* identifies different labels like names, dates, and other identifiers. However, these can be labeled differently across datasets (e.g., "email" in one dataset and "contact information" in another). This inconsistency makes it difficult to apply models trained with certain labels to another hospital that uses different labeling conventions. Additionally, Label definition issues complicate training. For example, the i2b2 2006 dataset (2.1.2) removes only ages over 89 years old, whereas the i2b2 2014 dataset (2.1.3) removes all ages. This discrepancy makes it impossible to use these two datasets together to determine the *age* label.

These challenges highlight the need for advanced NLP solutions that can handle the complexity and variability of healthcare data. The ultimate goal of this research is to develop methodologies that not only overcome these dataset limitations but also adapt to the evolving landscape of healthcare data management.

HIPAA-defined PHI Categories

To address the complexities of data consistency in healthcare de-identification, understanding the 18 categories of PHI defined by *HIPAA* is crucial. Table 1.1 lists these identifiers, which must be systematically recognized and anonymized by de-identification systems.

Table 1.1: The 18 categories of PHI as defined by *HIPAA*.

No.	Type of PHI
1	Names
2	Geographic subdivisions smaller than a state
3	Dates (except year) directly related to an individual
4	Phone numbers
5	Fax numbers
6	Email addresses
7	Social Security numbers
8	Medical record numbers
9	Health insurance beneficiary numbers
10	Account numbers
11	Certificate/license numbers
12	Vehicle identifiers and serial numbers, including license plate numbers
13	Device identifiers and serial numbers
14	Web Uniform Resource Locators (URLs)
15	Internet Protocol (IP) address numbers
16	Biometric identifiers, including finger and voice prints
17	Full face photographic images and any comparable images
18	Any other unique identifying number, characteristic, or code

1.3 Research Objectives

The research objectives outlined below are designed to bridge the gap between the theoretical potential of NLP technologies and their practical application in a diverse and dynamic healthcare environment.

1. **Comprehensive Evaluation of De-identification Techniques:** The first objective of this study is to perform a thorough evaluation of existing de-identification techniques, with a focus on their ability to handle unstructured data, language diversity, and label inconsistency. This evaluation will identify the strengths and weaknesses of current methods, pinpointing effective strategies that can significantly mitigate the mentioned challenges.

2. **Development of an Incremental Learning Annotator Tool (ILA):** The second objective aims at developing an annotation tool based on incremental learning principles. This tool will address the highlighted bottleneck of intensive labor and time required for manual dataset annotation, as detailed in the problem statement. The development of such a tool is intended to significantly lower the barriers to efficiently train and refine a model dedicated to whatever institution dataset with a focus on a "single dataset."

By fulfilling these goals, this research is intended to advance the field of clinical document de-identification, enhancing the privacy and utility of EHRs for broader research and clinical applications.

Chapter 2

Literature Review

2.1 Summary of Dataset Availability

Before examining the methodologies and metrics for de-identification in clinical documents, it is essential to provide an overview of the datasets used. Understanding their characteristics and statistics is crucial for contextualizing the subsequent analysis.

2.1.1 MedDocAn

The MedDocAn dataset [1] was delivered with a challenge composed of 2 tasks. The first task which concerns Named Entity Recognition (NER) contains a training set of 750 documents and a testing set of 250 documents, typically **Spanish** clinical case reports, crafted for the purpose of medical document anonymization.

Specifications:

- **Total Number of Documents:** 1000
- **Total Number of Tokens:** 424,127
- **Average Number of Tokens per document:** 424
- **Label Distribution:** The dataset contains 44 different labels defined with the BIO Format. In total, 14.50% of the tokens across the dataset are labeled.

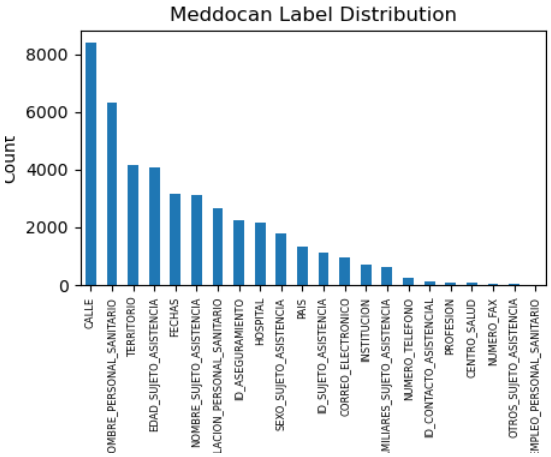


Figure 2.1: Bar graph of label distribution in the MedDocAn dataset.

[1]The MedDocAn dataset is available at this url: `https://github.com/PlanTL-GOB-ES/SPACCC_MEDDOCAN`

Pros and Cons in the Context of this Study

The number of documents in the MedDocAn dataset provides a solid foundation for training and assessing deep learning models. Its diversity of labels is beneficial and allow a mapping process to align MedDocAn's labels with the 18 PHI categories defined by HIPAA (1.1). This uniformization facilitates cross-dataset evaluation and allows effective transfer learning approaches.
The full mapping of the labels, the dropped labels and the supports are shown in Appendix A.1.1. [2]

Table 2.1: Comparison of F_1-Scores on MedDocAn dataset.

Method	F_1**-Score**	**Reference**
BERT-CRF	93.75	Mao et Liu [31]
BERT-CRF	**98.21**	Jiang et al. [22]
BERT-CRF	97.03	Marimon et al. [32]

2.1.2 i2b2 2006

The i2b2 (Informatics for Integrating Biology and the Bedside) 2006 dataset appeared with a challenge designed to de-identify clinical documents described by Uzeuner et al. [47]. The dataset, composed of patient discharge summaries, includes a training set of 669 documents and a testing set of 220 documents (24.75%).

The 2006 i2b2 challenge was one of the first **english** dataset to focus specifically on the task of de-identifying EHRs.

Specifications:

- **Total Number of Documents:** 889
- **Total Number of Tokens:** 554,997
- **Total Number of Labels:** 16
- **Average Number of Tokens per document:** 624
- **Label Distribution:** The dataset contains 16 different labels defined with the BIO-Schema. In total, 5.38% of the tokens across the dataset are labeled.

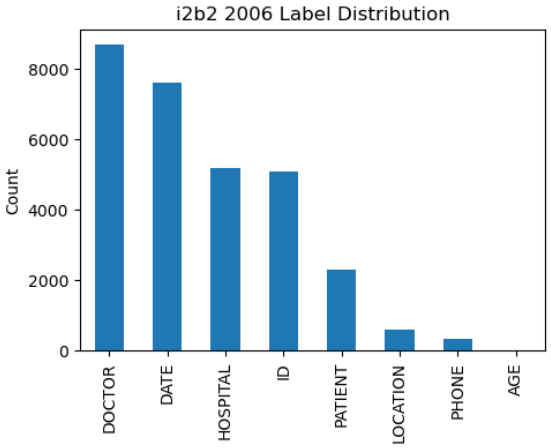


Figure 2.2: Bar graph of label distribution in the i2b2 2006 dataset.

[2]This table highlights the best studies for this specific dataset using the metrics described in 4.1.4, providing a clear and concise overview of the current state of the art performances on MedDocAn dataset. All metrics and methods presented in this and subsequent tables will be defined in detail later in the study.

Pros and Cons in the Context of this Study

The i2b2 2006 dataset is a valuable asset for this study on medical document anonymization. It's well-recognized in de-identification of clinical documents for its wide use in challenges. The full mapping of the labels, the dropped labels and the supports are shown in Appendix A.1.2.

Table 2.2: Comparison of F_1 scores on i2b2 2006 dataset

Method	F_1**-score**	**Reference**
Iterative Machine Learning	**99.75**	G. Szarvas et al. [42]
Bidirectional Transformers	98.45	Johnson et al. [24]
Rule-Based + ML	94.23	Lee et al. [27]

2.1.3 i2b2 2014

The i2b2 2014 dataset represents a continuation of the efforts initiated in 2006 to facilitate the development of de-identification methods for clinical documents. Accompanied by a challenge described by Stubbs et al. [41], the 2014 dataset is provided with 3 files respectively of 521, 269 and 514 documents. For this study, the third file will be used for the evaluation, representing 39.4% of the documents.

Specifications:

- **Total Number of Documents:** 1304
- **Total Number of Tokens:** 805,117
- **Total Number of Labels:** 46
- **Average Number of Tokens per document:** 617
- **Label Distribution:** The dataset contains 46 different labels defined with the BIO-Schema. In total, 4.45% of the tokens across the dataset are labeled.

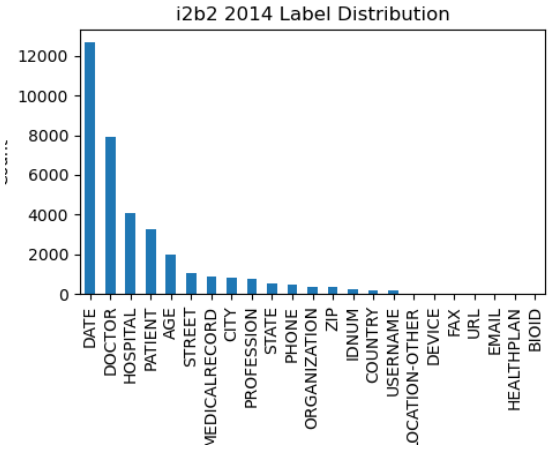


Figure 2.3: Bar graph of label distribution in the i2b2 2014 dataset.

Pros and Cons in the Context of the Study

The i2b2 2014 dataset is crucial for this study. It extends the i2b2 series by offering a diverse set of documents for de-identification. New document types and detailed annotations help test the algorithms effectively. Unlike the i2b2 2006 dataset, which mainly supported rule-based or

machine learning methods in the literature, the i2b2 2014 dataset came at a time when deep learning models were becoming popular. This timing provides many results and comparisons with other deep learning methods, making it very relevant and useful for this study. Therefore, this dataset is the most used in this research and will be the **baseline** dataset.

The full mapping of the labels, the dropped labels and the supports are shown in Appendix A.1.3.

Table 2.3: Comparison of F_1 scores for i2b2 2014 dataset

Method	F_1**-score**	**Reference**
Recurrent Neural Networks (RNNs)	97.85	F. Dernoncourt et al. [13]
Bi-LSTM-CRF with Neural Language Models	98.70	B. Tang et al. [44]
BERT - Transfer Learning	**98.90**	P.J. Chambon et al. [9]

2.1.4 multiNERD

The multiNERD dataset [45] offers a platform for evaluating the language capabilities of various models across a diverse linguistic landscape. This dataset is compiled from *Wikipedia* and *WikiNews* articles. The dataset is composed of 10 languages including English, French, Spanish, Dutsh and each language is composed of 2 files, one training file (±140880 documents) and one testing file (±16454 documents).

Specifications for multiNER English dataset:

- **Total Number of Documents:** 157,334
- **Total Number of Tokens:** 3,751,618
- **Total Number of Labels:** 31
- **Average Number of Tokens per document:** 24
- **Label Distribution:** Labels are defined with the BIO Format and include terms like *FOOD*, *PLANT*, *PERSON*. 12.544% of the tokens are annotated with these labels.

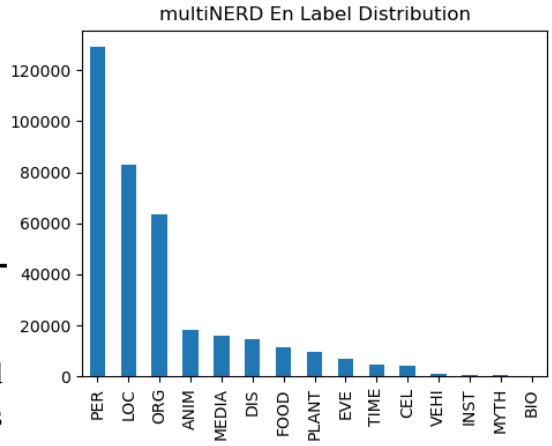


Figure 2.4: Bar graph of label distribution in the multiNERD English dataset.

Pros and Cons in the Context of this Study

The multiNERD dataset offers extensive data for evaluating language performance, but it lacks the medical specificity essential to this study. Despite its nonmedical nature and multilingual coverage,

it is valuable for assessing how well models trained on specialized datasets perform on general text and across different languages.

The full mapping of the labels, the dropped labels and the supports are shown in Appendix A.1.4.

Arsenault et al. [3] succeed to create a model that achieved an overall performance of 92.48% across the 10 different languages datasets.

2.1.5 wikiNER

The wikiNER dataset [37] is a multilingual dataset focused on de-identification annotations. In the context of this study, the focus will be exclusively on the **French** dataset. It contains few labels, but these labels are pertinent. The dataset is delivered with a train set composed of 123,364 documents and of a testing set of 10,728 documents (8.00%).

Specifications:

- **Total Number of Documents:** 134,092
- **Total Number of Tokens:** 3,499,679
- **Total Number of Labels:** 5
- **Average Number of Tokens per document:** 26
- **Label Distribution:** In total, 11.79% of the tokens across the dataset are labeled.

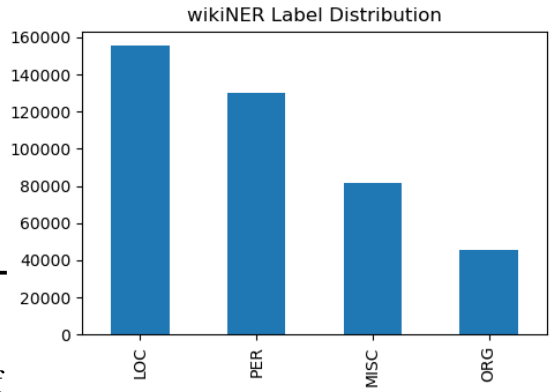


Figure 2.5: Bar graph of label distribution in the wikiNER dataset.

Pros and Cons in the Context of this Study

The documents are more general and not specific to medical content. This makes them suitable for studying transfer learning from general contexts. Although there are only five labels, but they can be easily mapped to the broader labels that will be defined later. This allows the dataset to be used for cross-validation with other datasets focused on de-identification tasks.

Rondeau and Su achieved an F_1 score of 89.28% using LSTMs. [3].

2.2 Historical Review of Different Approaches

Now that the datasets have been clearly defined, it is important to examine the various methodologies employed to achieve de-identification in clinical documents. This section will review different

[3]LSTMs stand for Long-Short Term Memory, describe in Appendix A.5.5

approaches, detailing their performance metrics. Particular focus will be given to deep learning methods and transfer learning solutions, highlighting their advancements and effectiveness in de-identifying sensitive information.

2.2.1 Human Annotation

Human annotation for de-identification, as described by Carrell et al.[8], involves manually checking clinical notes to find and hide PHI. In their study on 100 clinical notes from family practice clinics, they found a median recall of **96.6%** with one annotator, increasing to **99.5%** with a second annotator, at an average cost of $30.16 per document or $2.76 per PHI instance with four annotators.

In another study, Jonnagaddala et al.[25] confirmed these results with an inter-annotator agreement (IAA) of **94.6%** on the OpenDeID corpus, which includes 2,100 pathology reports. They also compared annotators working independently to those using pre-annotations, finding that pre-annotations **reduced time** but slightly **lowered recall rates**.

Grouin and Névéol [18] noted that using rule-based systems for pre-annotations can improve annotation accuracy and time efficiency, achieving F-measures between **97%** and **99%**.

These findings highlight the balance between annotation speed, annotation quality and the need for efficient pre-annotation systems to assist human annotators with large datasets.

Table 2.4: Performance metrics of human annotation

Method	**Avg Time/Doc (min)**	**Avg Time/Token (sec)**	F_1 **Score** (%)
Carrell et al. [8]	9.83	1.59	**99.5**
Jonnagaddala et al. Serial [25]	3.21	0.26	95.52
Jonnagaddala et al. Pre-Annotation	2.38	**0.19**	84.55
Grouin et Névéol [18]	**2.04 to 3.2**	0.42 to 0.65	97.00 to 98.70

Manual annotation, while time-consuming and expensive, provides high-quality results, especially with multiple annotators. However, due to the expensive cost, manual annotation alone is not practical for de-identifying all documents. Using pre-annotations can improve accuracy and reduce time, but it requires careful revision. Thus, a good approach is to use manual annotation with pre-annotations to create a high-quality gold standard corpus. This corpus can then train machine learning or deep learning models to de-identify more documents.

2.2.2 Rule-Based

Rule-based systems, first introduced by Gupta et al. in 2004 [20], are foundational in clinical document de-identification. These systems use predefined rules crafted by experts and **do not require annotated training data**. They employ dictionaries, regular expressions (regex), and manually written rules that need regular updates to match evolving language and clinical terminology.

Key Performances

While studies have demonstrated high recall, such as Beckwith's achieving 98% and Friedlin's achieving 99% [5, 17], the precision of these tools is often lacking. For instance, Beckwith's study reported a precision of only 42%, and even the state-of-the-art Neamatullah tool shows a precision of just 75% [35]. For detailed reviews, refer to [34, 36].

Conclusion and Limitations

Rule-based systems rely on predefined rules to anonymize PHI in clinical documents. A significant challenge is handling textual ambiguity, which reduces precision. For example, the term "Parkinson" could refer to "Parkinson's disease" or a physician named "Dr. Parkinson," leading to errors in de-identification. Despite their high recall in both structured and unstructured datasets, their low precision is a major drawback. This necessitates improvements through the integration of machine learning techniques, which aim to enhance the adaptability and accuracy of these tools, better addressing the complexities of unstructured texts.

2.2.3 Machine Learning and Hybrid System

Since 2006, following Aramaki et al. [2], de-identifying PHI in clinical documents has significantly evolved from rule-based systems to advanced machine learning (ML) techniques, enhancing the accuracy and efficiency of automated PHI privacy protection.

Initially, rule-based systems were favored for their straightforward implementation but struggled with natural language nuances. ML approaches, learning from large datasets, offer better flexibility and **contextual understanding**, essential for handling ambiguous terms. A detailed description of different ML techniques can be found in Appendix A.3.

Hybrid Approach and Performance

Hybrid systems combining rule-based and ML methodologies address the limitations of each approach. Leveraging the robust recall of rule-based systems and the precision of ML models, it improves de-identification processes. Notable implementations include Dehghan et al. [12] and Liu et al. [29], integrating Conditional Random Field (CRF) models with rule-based components.

Table 2.5: (Left) Performance of different machine learning and hybrid approaches. (Right) Precision and recall on cross-dataset setup, based on Ferrandez et al. [16].

Study	**Method**	F_1**-Score**
Tafvizi et al. [43]	SVM	95.41
Aramaki et al. [2]	CRF	97.47
Szarvas et al. [42]	Decision Trees	**99.75**
Dehghan et al. [12]	CRF + Rules	94.82
Liu et al. [29]	CRF + Rules	94.83

	Method	**Precision**	**Recall**
Rule-Based	HMS Scrubber	23	47
	MeDS	34	**60**
	MIT deid	76	**60**
ML	MIST	**90**	44
	HIDE	84	46

Despite impressive performance, these systems rely on extensive, well-annotated datasets, which are costly to compile. They perform best on data similar to their training sets, with reduced effectiveness on new or different clinical data. Ferrandez et al. [16] note significant performance drops when systems are applied to unfamiliar datasets, with maximal recall of 46% for Machine Learning systems and 60% for Rule-Based approaches.

2.2.4 Deep Learning

Traditional rule-based methods provide poor precision in PHI de-identification, while machine learning methods, though offering better performance, require large, well-annotated datasets and perform poorly on unseen data (see Table 2.5). These limitations are costly and hinder scalability, particularly in healthcare where document variability is high.

Deep learning models, however, excel by extracting and generalizing features from diverse datasets, effectively processing complex clinical texts. Transformers [4], introduced by Vaswani et al. [48], revolutionized Natural Language Processing (NLP) by eliminating the need for sequential data processing.

All methods used in Deep Learning, from RNNs to Transformers, along with their main differences and improvements, are described in Appendix A.4.

Table 2.6: Performance of different deep learning approaches in de-identification tasks.

Study	**Method**	F_1**-Score (%)**
Jiang et al. [23]	CRF + LSTM	85.7
Dernoncourt et al. [13]	CRF + ANN	97.8
Johnson et al. [24]	BERT-large	**98.8**
Ahmed et al. [1]	Self-Attention	98.2
Meaney et al. [33]	ROBERTA	96.7

[4]The "Attention Is All You Need" paper and the introduction of Transformer architectures have been instrumental in the emergence of large language models (LLMs). This groundbreaking technology paved the way for the development of advanced models like ChatGPT, Gemini, Claude and many others, sparking a competitive race to develop the most capable and versatile LLMs.

2.2.5 Transfer Learning

Off-the-shelf or naive transfer learning: This approach involves taking a model fine-tuned on another dataset and directly evaluating it on a new, unseen dataset without any additional customization. The advantage is that it does not require annotated datasets for the new domain, but it often results in lower performance compared to customized approaches.

Fully customized systems: In this approach, a model that has undergone naive transfer learning is further fine-tuned on the target dataset, which is the dataset on which the model will be used for prediction. This requires some annotated documents but generally leads to higher accuracy as the model parameters are adjusted to better fit the specific characteristics of the new data.

Johnson et al. [24] used a BERT-based model pre-trained on general language data and fine-tuned on several clinical datasets. The table below shows the F_1-scores from this study, illustrating the model's performance across different datasets for the Naive Transfer learning approach.

Table 2.7: Naive Transfer learning evaluation of a BERT-based model: F_1-scores for models trained on one dataset and tested on unseen datasets.

Trained/Tested on	i2b2 2014	i2b2 2006	PhysioNet	Dernoncourt-Lee
i2b2 2014	**98.62**	81.62	87.95	88.32
i2b2 2006	92.77	**98.45**	75.37	86.85
PhysioNet	83.84	52.28	**95.61**	78.54
Dernoncourt-Lee	84.02	63.13	90.27	**97.42**

The table shows that models trained on one dataset do not perform well on unseen datasets, highlighting the importance of fine-tuning for specific data sources. For example, the model trained on i2b2 2014 achieved high performance on the same dataset (98.62) but performed significantly worse on i2b2 2006 (81.62) and PhysioNet (87.95).

Hartman et al. [21] compared off-the-shelf models with fully customized models.

- Off-the-shelf models generally achieved accuracies above 90%.
- Fully customized systems achieved de-identification accuracies between 97% and 99%.

Studies have shown that naïve transfer learning is infeasible due to catastrophic performance. However, transfer learning holds significant potential. The potential of transfer learning, particularly the number of documents required to fine-tune the transfer model, will be analyzed in a subsequent section.

CHAPTER 3

THEORETICAL FOUNDATIONS

This chapter will focus on technical aspects such as Transformers, specifically the BERT architecture, which is the only Transformer architecture used in this study. For a thorough understanding of the theoretical foundations of deep learning, including perceptrons, activation functions, RNNs, LSTMs, and self-attention mechanisms, refer to Appendix A.5. The appendix provides a comprehensive overview of deep learning models and their application to NER tasks, highlighting their strengths and challenges.

3.1 Tokens and Tokenizers

Tokens are the fundamental units of text, which can be words, subwords, or characters, processed by models during NLP tasks. Tokenization is the process of breaking down text into these basic units.

Tokenizers are essential for preparing text for models as they convert raw text into tokens.

This study consistently uses the tokenizer that corresponds to the model in use. Tokenizers produce several key outputs, including:

- **Token IDs**: Numerical representations of each token in the vocabulary.
- **Attention Masks**: Binary masks indicating which tokens are actual content (1) and which are padding (0).
- **Word IDs**: Identifiers for each original word before tokenization, useful for mapping tokens back to words.
- **Offsets**: The start and end positions of tokens in the original text, aiding in aligning tokens with their positions in the input text.

These outputs ensure that the model correctly interprets and processes the text data, maintaining alignment with the original input structure.

3.2 Transformer Models

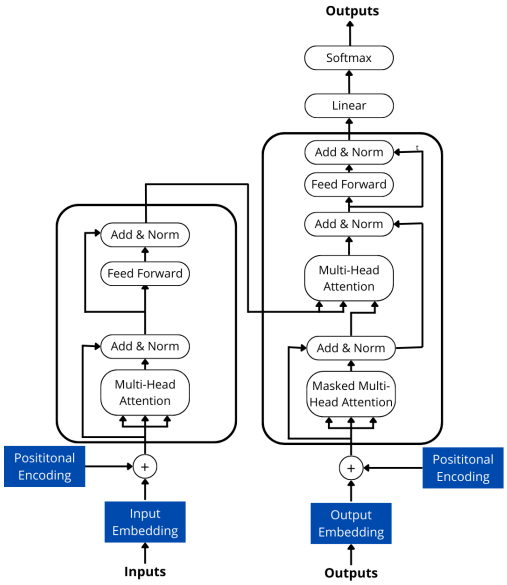


Figure 3.1: Transformer architecture

Transformers revolutionize NLP by using a self-attention mechanism that processes sequences in parallel, enhancing efficiency and performance. The architecture comprises two main components: the encoder on the left and the decoder on the right. The encoder transforms the input sequence into a contextual representation through multiple layers, each containing a multi-head self-attention mechanism and a feed-forward neural network. This setup allows each token to attend to all others, capturing dependencies regardless of position. In practice, this means that the first word in a sentence can be as closely linked to the last word as any other, providing a deep and holistic understanding of the context within the entire sequence.

The self-attention mechanism involves creating three vectors for each token: Query (Q), Key (K), and Value (V). These vectors are computed as follows:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V$$

where X is the input token embedding and W^Q, W^K, and W^V are learned weight matrices. The attention scores are calculated by taking the dot product of Q and K, scaling by the square root of the dimension of K, and applying a softmax function:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Multi-head attention involves running several self-attention operations in parallel, allowing the model to capture different aspects of the relationships between tokens. The outputs are then concatenated and linearly transformed.

The decoder, similar to the encoder, includes an extra attention layer that focuses on the encoder's output. It generates sequences autoregressively, predicting one token at a time while attending to previously generated tokens using masked multi-head attention.

Positional encodings are added to the input embeddings to retain the order of the sequence, as the self-attention mechanism itself does not inherently capture positional information.

Masked multi-head attention ensures the decoder only attends to tokens generated so far, maintaining the autoregressive property. This combination of masked attention and positional

encodings allows the decoder to process tokens in parallel, leveraging GPU efficiency. Unlike RNNs, which process tokens sequentially, the transformer decoder can handle multiple tokens at once, making the process faster and more efficient.

3.3 *BERT* Model

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained model that captures the context of words by considering their surrounding words in both directions. It is pre-trained using two tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). In MLM, 15% of tokens are randomly masked, and the model learns to predict them, helping it understand context deeply. NSP involves predicting if one sentence follows another, aiding in understanding sentence relationships.

For fine-tuning, *BERT* is further trained on specific tasks with labeled data, adapting the pre-trained model to various NLP applications (NER for Named Entity Recognition is the task concerned in this study).

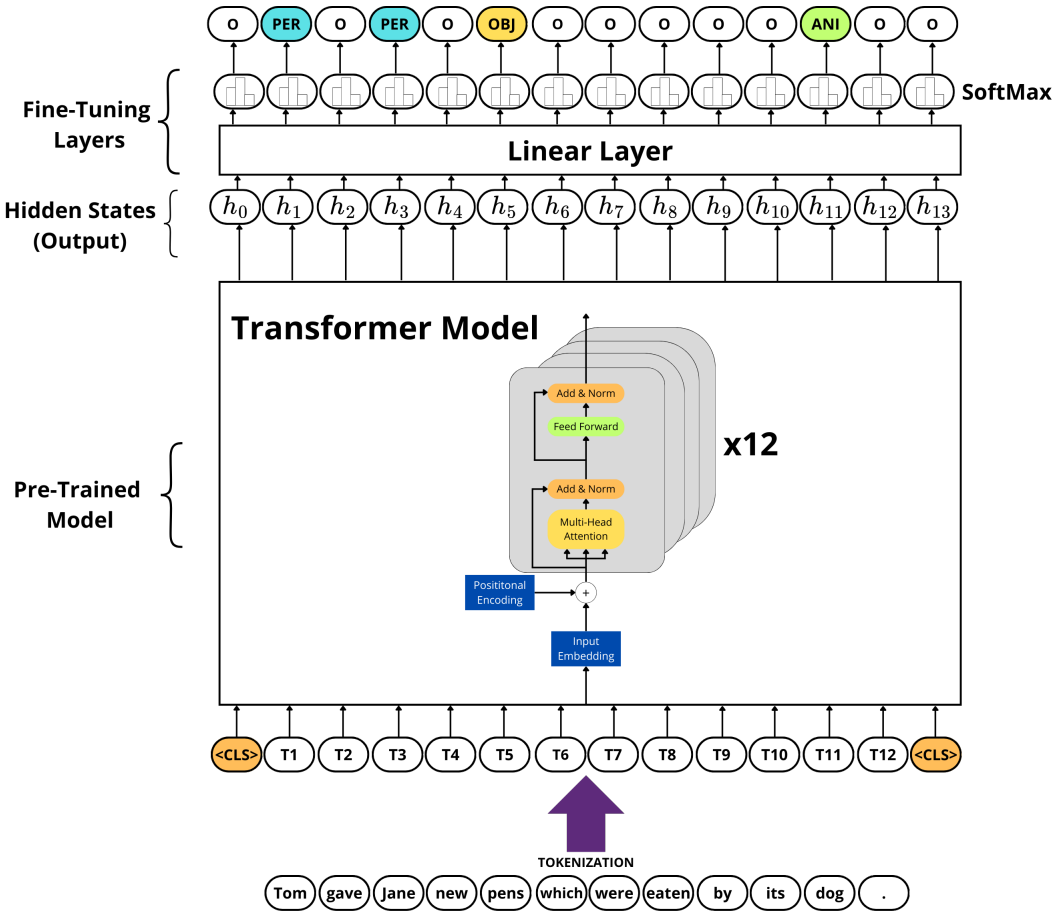


Figure 3.2: *BERT* model architecture with NER Task fine-tuning process

Initially, the input text is tokenized and embedded with positional encodings to preserve the order of the sequence. These embeddings are then processed through multiple layers of the transformer model. The output, known as hidden states, is used by task-specific layers.

For NER, this involves a linear layer followed by a softmax function, which assigns probabilities to each token for different labels and selects the most likely label for classification. For simplicity, the step of converting tokens back to their original text form is not shown in the figure.

3.4 *BERT* Justification

BERT-base architecture is chosen as transformers architecture in this study due to several key advantages:

- **Lightweight**: Some *BERT* architectures are lightweight and can run efficiently on various hardware, making them versatile for different deployment environments.
- **Variants of BERT**: There are numerous *BERT* variants such as *CamemBERT*, *RoBERTa* or *BioBERT*. This variety is beneficial as it allows for consistent experimentation across different settings such as different languages, domains or pre-training data..
- **Open-source and Accessibility**: *BERT* is open-source and easily accessible through platforms like *Hugging Face*, facilitating integration and experimentation.
- **Well-documented**: *BERT* is well-documented for the NER task, particularly in clinical de-identification, enabling robust comparison and validation of results.
- **High Performance**: *BERT* has demonstrated high performance in NER tasks, making it a reliable choice for accurate and efficient entity recognition in clinical texts. Some studies like Polignano et al. [38] or Lothritz et al. [30] demonstrated that *BERT*-based models perform best than other transformers architecture like *XLNet* or *ELECTRA*.

3.5 Training Parameters

The effectiveness of fine-tuning these models depends on several key parameters. An epoch refers to the number of complete passes through the training dataset, allowing the model to iteratively adjust its weights and improve performance over time. The right number of epochs ensures adequate learning without overfitting.
Batch size, the number of training examples per iteration, influences training stability and speed. Larger batch sizes lead to more stable gradient estimates and faster convergence but require more memory. Selecting an appropriate batch size balances training efficiency and resource usage.

The learning rate controls the step size taken by the optimizer to minimize the loss function. A well-tuned learning rate is crucial for efficient training, affecting both convergence speed and the ability to find an optimal solution. Proper adjustment is key to optimal training efficiency and accuracy.

3.6 Incremental Learning

Incremental learning enables models to update with new data continuously, without the need for full retraining. This method ensures efficiency and scalability. In NER, incremental learning allows models to stay up-to-date with new entities and terminology. By continually learning from fresh data, models maintain high performance and accuracy, adapting quickly to evolving information.

3.7 BIO-Tagging

BIO-tagging, also known as IOB-tagging, is a method for labeling tokens in a text to identify the boundaries of entities. Each token is tagged as Beginning (B), Inside (I), or Outside (O) of an entity.

```
John B-PER
Doe I-PER
was O
born O
on O
June B-DATE
31 I-DATE
, I-DATE
2000 I-DATE
. O
```

Listing 3.1: BIO-tagging

```
John PER
Doe PER
was O
born O
on O
June DATE
31 DATE
, DATE
2000 DATE
. O
```

Listing 3.2: Non-BIO-tagging

In the BIO-tagging example, "John" is tagged as B-PER (beginning of a person's name) and "Doe" as I-PER (inside a person's name). Dates are similarly marked, with "June" as B-DATE (beginning of a date) and subsequent parts of the date as I-DATE (inside a date). In contrast, the non-BIO-tagging example labels entire entities without indicating the boundaries between the beginning and inside of entities.

3.8 Frameworks and Tools

Frameworks like *Hugging Face* Transformers library provide access to pre-trained models and simplify the fine-tuning process.

```
from transformers import AutoModelForTokenClassification, AutoTokenizer
model = AutoModelForTokenClassification.from_pretrained("bert-base-uncased")
tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
```

This snippet demonstrates loading a pre-trained *BERT* model and tokenizer.

Chapter 4

Evaluation of Current De-Identification Methods

4.1 Methodology

This section outlines the methodologies employed in evaluating the de-identification methods. The foundation of any robust evaluation lies in the clarity of the dataset format used, the metrics by which success is measured, and the baseline model for comparison. All the code of this section is available here: `https://github.com/NicolasHuberty/DeepDeId-ClinicalDocs/tree/evaluation`.

4.1.1 Dataset Format Used

In the context of de-identification in clinical documents, many formats of annotation exist. The most used formats are detailed in Appendix A.6.1.

The *tsv* format was chosen for this work. This simplifies data annotation by aligning each word with its corresponding label in a tab-separated values format. Each document is contained on the same *tsv* file, separated by \n.

```
word label
word label
...
\n
word label
...
```

To accommodate these diverse formats, conversion functions were implemented for each existing format in the medical field, standardizing them into the *tsv* format.

4.1.2 Baseline Experimental Setup

Various setups and methods will be examined during the evaluation. To ensure consistency across different evaluations, a baseline setup will be established that will remain constant unless specific changes are being assessed.

Model Configuration

The baseline model is a custom instance of *RoBERTa* (details of this custom model can be found in Appendix A.2). This custom model includes a specially designed linear layer to meet specific classification needs. The imported *RoBERTa* model from *Hugging Face* comes with a predefined linear layer containing existing labels such as MISC or ORG, which do not align with the requirements. Therefore, a custom linear layer was implemented to ensure the model accurately fits the intended classification criteria. The Python code snippet below illustrates how the model and tokenizer are called:

```
tokenizer =
    RobertaTokenizerFast.from_pretrained("Jean-Baptiste/roberta-large-ner-english")
model = RobertaCustomForTokenClassification(num_labels=num_labels)
```

The decision to use the *RoBERTa* model is based on its superior performance compared to *BERT* in NER tasks. *RoBERTa* achieves this through extensive optimization techniques, including training on larger datasets, longer training times, and removing the next sentence prediction objective found in *BERT*. These optimizations result in more robust contextual representations and improved generalization capabilities. Several studies have demonstrated that *RoBERTa* consistently outperforms *BERT* in various NER benchmarks, highlighting its effectiveness and efficiency. For instance, studies such as Zhong et Goodfelllow [49] provide empirical evidence of *RoBERTa*'s enhanced performance over *BERT*, making it a more suitable choice for tasks requiring high accuracy in entity recognition.

Label Definition

To maintain consistency between different evaluations, two strategies were defined for label mapping: keeping labels specific to datasets when possible and grouping labels when necessary (such as in a transfer learning setup).

- To evaluate model performance on **specific datasets**, the native labels of each dataset were used, excluding the BIO-Format (also called IOB-Tag).
- To compare analyses across **different datasets**, labels were merged into broader categories to standardize and simplify comparisons. The broader categories used are: PERSON, LOCATION, DATE, and ID.

Training Parameters

The model was trained using a batch size of 4 to optimize the balance between memory usage and performance. Additionally, the training was conducted over 5 epochs to ensure sufficient learning without overfitting, and a learning rate of $5e^{-5}$ is used.

4.1.3 Loss Function

The loss function measures the discrepancy between the model's predictions and the actual data. For the de-identification task, **Cross-Entropy Loss** is utilized, as it is particularly well-suited for models that output a probability distribution across multiple classes.

The formula for Cross-Entropy Loss for a multi-class classification problem is given by:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{4.1}$$

Here, y is the binary indicator (0 or 1) if class label i is the correct classification for the observation, C is the number of classes, and $\hat{y}$ is the predicted probability that the observation is of class i.

The use of Cross-Entropy Loss ensures that the model learns to assign high probabilities to the correct class and low probabilities to all other classes, which is aligned with the goal of minimizing false negatives and false positives in PHI identification.

4.1.4 Evaluation Metrics

De-identification models are evaluated based on their ability to accurately identify and classify PHI in clinical documents. To quantify model performance, a confusion matrix and several derived metrics are utilized.

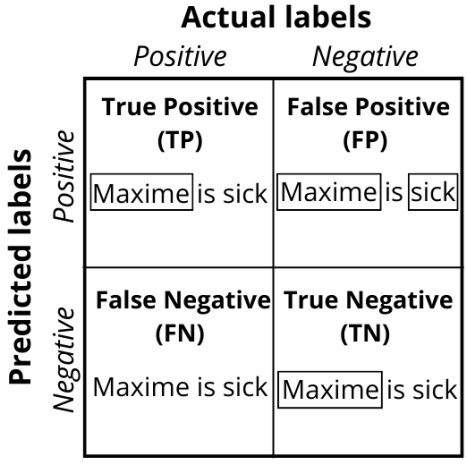


Figure 4.1: Confusion Matrix example with NER Task.

Confusion Matrix Explanation:

- **True Positives (TP)**: Correctly identified PHI.
- **False Positives (FP)**: Non-PHI incorrectly labeled as PHI.
- **True Negatives (TN)**: Correctly identified non-PHI.
- **False Negatives (FN)**: PHI incorrectly labeled as non-PHI.

This matrix helps understanding the trade-offs between different types of errors and correct decisions made by the model.

The effectiveness of the model is evaluated using several metrics derived from the confusion matrix:

- **Precision (Positive Predictive Value)** – Precision quantifies the accuracy of positive predictions made by the model. It is crucial in scenarios where false positives carry significant consequences, such as misclassifying non-PHI data as PHI. The formula for calculating precision is given by:

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

- **Recall (Sensitivity)** – This metric assesses the model's ability to correctly identify all PHI instances in the dataset. A high recall rate is essential to minimize the risk of exposing sensitive data. Recall is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

- F_1 **Score** – The F_1 score combines both precision and recall into a single measure, balancing both aspects for a more comprehensive model evaluation. For multi-class classification, two averaging methods are used:

  - **Macro Average F_1:** The Macro Average calculates the F_1-score independently for each class before averaging these scores. This method emphasizes the performance of each class equally, regardless of its frequency within the dataset. It is particularly strict on any class that performs poorly, as each class's score impacts the overall average equally. This approach is beneficial for ensuring that models perform well across all categories, not just the most prevalent ones. The macro average, **adopted as the standard in this study**, is defined as:

$$Macro\text{-}F_1 = \frac{1}{N} \sum_{i=1}^{N} F_{1i} \tag{4.4}$$

  - **Micro Average F_1:** The micro average aggregates the contributions from all classes to compute a single metric, useful in cases of class imbalance. It is calculated using:

$$Micro\text{-}F_1 = 2 \times \frac{\sum TP \times \sum Recall}{\sum TP + \sum Recall} \tag{4.5}$$

4.1.5 Evaluation Methods: Entity-Based vs. Token-Level

In NER tasks, evaluation methods are essential for assessing model performance. These methods can be divided into entity-based and token-level approaches, each with distinct advantages.

Entity-Based Evaluation: This approach requires exact matches between predicted and annotated entities, including accurate start and end positions. This method ensures that entities are

identified precisely and contextually intact, which is critical for applications needing strict boundary adherence.

Token-Level Evaluation: This approach evaluates the model based on the accurate classification of individual tokens, regardless of entity boundaries. It is more flexible, particularly useful for datasets where entities are not clearly delineated or when the goal is to identify all potentially sensitive information.

For this de-identification project, **token-level evaluation** is preferred due to its flexibility and robustness in handling annotation inconsistencies.

Figure 4.2: Comparison of Entity-based and Token-Level Evaluation

This figure illustrates a scenario encountered in one of the i2b2 challenges. In the annotated dataset, the dates "2001-11-16" and "November 16, 2001" are marked as a single entity. However, the model identifies these dates as separate entities due to the format differences.

In an entity-based evaluation, this would be considered as a misclassification since it expects the entire span to be recognized as one continuous entity. As shown in the figure, the boundaries are not respected in the model's two separate predictions. Conversely, a token-level evaluation recognizes the correct identification of each individual date component, regardless of the boundaries. This is crucial for ensuring all PHI elements are accurately detected, even if the model's predictions do not align perfectly with the entity boundaries.

This token-level approach aligns more closely with the de-identification goals, which prioritize the accurate detection of all PHI. By focusing on individual tokens, the model ensures that all sensitive information is identified, even if the format differs. Therefore, token-level evaluation has been chosen for this study to better meet the objectives of accurate and comprehensive de-identification.

4.2 Evaluation of Traditional Approaches

This section outlines the evaluation process for various traditional methodologies applied to the problem statement, structured in two distinct phases. Initially, the evaluation focuses on single dataset analysis to determine optimal performance metrics within individual datasets. Subsequently, the study advances to cross-dataset evaluations, aimed at assessing the robustness and consistency of these methodologies across different datasets. This dual-phase approach ensures a comprehensive assessment of each method's effectiveness in addressing the research objectives.

4.2.1 Precise Results of *RoBERTa* on i2b2 2014

First, a detailed result showcasing specific performances of the baseline model on the baseline dataset is presented. This result demonstrates the strong performance achieved when working with a single dataset. The table offers a comprehensive depiction of *RoBERTa*'s results on the i2b2 2014 dataset. For future work, the F_1-score will be the sole metric used to define performance results. The table highlights promising outcomes, with few false negatives, which is critical for the context of this study.

Table 4.1: Performance metrics for i2b2 2014 with *RoBERTa*

Label	**Support**	**TP**	**FP**	**FN**	**TN**	**Precision**	**Recall**	**F_1-Score**
HOSPITAL	1143	1113	49	30	152988	95.8	97.4	96.6
STREET	337	337	3	0	153840	99.1	100.0	99.6
MEDICALRECORD	314	313	2	1	153864	99.4	99.7	99.5
AGE	461	459	7	2	153712	98.5	99.6	99.0
ZIP	112	112	0	0	154068	100.0	100.0	100.0
STATE	151	142	3	9	154026	97.9	94.0	95.9
PHONE	122	120	4	2	154054	96.8	98.4	97.6
COUNTRY	79	67	1	12	154100	98.5	84.8	91.2
USERNAME	26	25	0	1	154154	100.0	96.2	98.0
CITY	216	210	25	6	153939	89.4	97.2	93.1
IDNUM	22	20	3	2	154155	87.0	90.9	88.9
DATE	2561	2537	9	24	151610	99.6	99.1	99.4
PROFESSION	124	118	8	6	154048	93.7	95.2	94.4
PATIENT	1067	1060	2	7	153111	99.8	99.3	99.6
ORGANIZATION	60	34	0	26	154120	100.0	56.7	72.3
DOCTOR	1274	1257	17	17	152889	98.7	98.7	98.7
O	146098	146036	87	62	7995	99.9	100.0	99.9

4.2.2 Label Definition

One of the first aspects to assess in this study is the performance of the labeling schema, specifically the use of the BIO schema versus a non-BIO schema. While the BIO schema is valuable for marking entity boundaries and enhancing NER by improving the model ability to detect and segment PHI accurately, its necessity may diminish when focusing on token-specific evaluation, where the classification of each word is crucial.

Additionally, the performance of the grouping schema must be evaluated, which is pivotal for evaluating transfer learning. In this schema, PHI types can either be treated as separate categories

or grouped into broader categories. For example, geographical identifiers such as countries, cities, and streets can be grouped under a single 'LOCATION' label. This approach not only helps in addressing inconsistencies across different datasets but also facilitates the application of transfer learning techniques, enabling comparisons and integrations across datasets that originally used disparate labeling systems.

Results

The following tables summarize the model performance on the *i2b2 2014* dataset under different labeling and schema conditions. The same tests have been evaluated on the *Meddocan* dataset (in Appendix A.6.2 and confirm the results.

Table 4.2: BIO-Schema Performance on i2b2 2014 Dataset (Individual Labels)

Label	Support	BIO F_1	~~BIO~~ F_1
DOCTOR	1274	97.9	**98.4**
PATIENT	1067	98.4	**99.0**
DATE	2561	98.6	**99.3**
COUNTRY	79	84.7	**89.0**
CITY	216	91.6	**93.1**
STREET	337	98.1	**98.8**
ZIP	112	100.0	100.0
STATE	151	93.7	**98.4**
HOSPITAL	1143	**96.2**	95.7
MEDRECORD	314	99.4	99.4
PHONE	122	95.6	**97.9**
O	146098	99.9	99.9

Table 4.3: BIO-Schema Performance on i2b2 2014 Dataset (Grouped Labels)

Label	Support	BIO F_1	~~BIO~~ F_1
PERSON	2341	98.5	**99.3**
DATE	2561	98.8	**99.3**
LOCATION	2106	97.1	**97.8**
ID	485	98.2	**99.0**
O	146098	99.9	99.9

Table 4.4: Individual vs. Grouped Label Mapping on the *i2b2 2014* Dataset

Label	Support	Indiv. F_1	Grouped F_1
PERSON	2341	98.7	**99.3**
DATE	2561	99.3	99.3
LOCATION	2106	96.1	**97.8**
ID	485	98.5	**99.0**
O	146098	99.9	99.9

These tables provide a clear comparative analysis of model performances. The results are unequivocal and demonstrate that the baseline setup is superior without the BIO schema. Furthermore, the models benefit from using grouped labels. This is consistently shown in MedDocAn experiment, showing that models perform better with fewer, broader label categories.

4.2.3 Language Performances

Now that the label definitions have been evaluated, another crucial aspect from the problem statement is addressed.: the impact of language diversity on performance. The primary goal is to assess

how language affects model performance to determine the best approach for tackling this problem. Specifically, the aim is to find out whether a single multilingual model can perform better than individual models for each language, or if specialized language-specific models are necessary.

Results

Table 4.5: Model-Specific Language Impact: Performance comparison of models

Label	**MEDDOCAN**			**i2b2 2014**			**multiNERD Fr**		
	mBERT	*sp*	*en*	*mBERT*	*sp*	*en*	*mBERT*	*sp*	*en*
PERSON	99.15	99.86	**99.88**	98.42	97.78	**99.34**	**97.91**	94.38	96.06
LOCATION	99.00	98.85	**99.29**	95.69	95.76	**97.85**	**93.90**	86.20	88.34
DATE	99.48	99.48	**99.69**	99.07	98.26	**99.26**	**70.10**	67.34	58.92
ID	**99.47**	99.26	99.15	98.69	98.62	**98.97**	-	-	-
O	99.94	99.93	**99.95**	99.91	99.87	**99.95**	**99.42**	99.14	99.22

Table 4.6: Performance metrics for different models across various languages

Label	**RoBERTa**				**mBERT**				**BETO**			
	Fr	Nl	En	Es	Fr	Nl	En	Es	Fr	Nl	En	Es
EVE	84.4	84.0	**89.5**	88.1	87.4	**87.5**	85.6	81.4	76.5	65.0	72.2	**81.3**
BIO	0.0	9.5	0.0	**15.8**	0.0	15.8	0.0	**48.0**	0.0	7.7	0.0	**42.3**
MEDIA	70.3	79.0	**93.2**	75.9	67.0	73.5	**86.9**	75.6	55.6	53.9	**87.7**	76.4
TIME	63.9	71.0	71.5	**71.8**	68.0	71.0	56.9	**72.8**	60.0	60.3	34.5	**70.5**
ORG	82.0	87.7	**96.8**	91.3	81.7	89.5	**91.7**	88.9	57.3	64.6	80.7	**86.6**
PLANT	26.4	21.8	**59.7**	28.0	35.0	39.4	**51.2**	46.1	25.5	18.0	39.6	**41.7**
LOC	87.0	94.1	**98.8**	94.3	89.9	**95.6**	95.0	93.0	70.4	77.2	86.7	**92.0**
ANIM	24.6	53.6	**69.7**	40.3	43.0	**66.5**	61.6	49.1	21.0	**50.7**	47.8	49.8
PER	95.7	96.9	**99.3**	97.5	97.5	98.4	**99.1**	98.5	88.7	88.1	95.8	**97.4**
CEL	9.8	30.9	**62.1**	23.2	54.3	**65.2**	55.6	64.2	15.9	24.5	31.7	**61.2**
MYTH	**5.5**	0.0	4.9	4.7	8.8	33.9	14.3	**46.8**	15.3	1.3	0.0	**27.7**
VEHI	44.3	37.4	**70.2**	64.3	48.5	49.6	51.5	**53.0**	30.4	**48.3**	29.9	22.5
INST	**46.6**	0.0	0.0	42.5	49.9	0.0	0.0	**54.3**	19.1	0.0	0.0	**54.2**
DIS	42.8	38.3	**69.4**	52.7	53.8	45.7	**61.5**	57.9	35.8	26.6	**55.4**	55.1
FOOD	4.3	24.6	**50.5**	30.0	19.7	32.7	**39.9**	33.2	1.0	7.9	31.4	**34.3**
O	98.9	98.6	**99.3**	98.8	98.7	98.6	**98.9**	98.6	97.7	97.2	98.2	**98.5**

For this table, the first 1,000 training documents of the multiNERD dataset (2.1.4) were utilized, which provides a unique advantage by offering identical data content across multiple languages. This allows direct comparisons under controlled linguistic variations.

Below are the configurations for the different models that are not on the *baseline* setup:

mBERT (Multilingual BERT):[14] This model is designed to handle multilingual data and has been trained on a diverse set of languages. The initialization code can be found in Appendix A.6.7.

sp (Spanish BERT - BETO):[7] Specifically tailored for Spanish language data, this model leverages a Spanish-specific tokenizer and classification model. The initialization code can be found in Appendix A.6.8.

The results of the first table show that the English model performs very well, even on non-English datasets. For example, it outperforms the Spanish model on the Meddocan dataset. Additionally, the *mBERT* model performs well on three datasets and surpasses other models on the French dataset.

The second table indicates that models perform best on their respective languages. *RoBERTa* (English) and *BETO* (Spanish) achieve the highest results for their languages. Again, the *mBERT* model shows no language preference and delivers promising results.

4.2.4 Impact of the model

This section aims to achieve the best possible performance by evaluating various models to determine which performs best on clinical documents. This assessment will help to identify the most effective model for the specific application in the clinical domain. The focus centers on three critical aspects:

1. **Domain-Specific Models versus General Models:** The evaluation investigates whether domain-specific models, such as *PubMedBERT* which is tailored for biomedical text, outperform general language models. This comparison is crucial, given the specialized terminology and context used within clinical datasets.

2. **Impact of Model Variants:** This aspect assesses how different variants of the same model (e.g. base vs. large, cased vs. uncased) affect performance on medical datasets. It explores which variants best capture the nuances of medical language and terminology, potentially influencing the model selection for further applications.

3. **Comparative Analysis of Leading Models:** The study compares top-performing models, including multilingual and monolingual configurations, to identify which models consistently deliver superior performance across diverse medical datasets. This comparison helps in understanding if there is a universally superior model or if the optimal choice depends on specific dataset characteristics.

Results

Table 4.7: Performance Comparison of general model compared to domain-specific model

Label	**i2b2 2006**				**i2b2 2014**			
	rob	*pm*	*sci*	*bio*	*rob*	*pm*	*sci*	*bio*
PERSON	**97.91**	95.47	96.18	94.90	**99.34**	98.64	98.85	97.77
LOCATION	**97.51**	96.06	94.92	96.96	**97.85**	96.34	96.64	95.20
DATE	99.11	98.97	99.21	**99.38**	99.26	**99.28**	99.16	98.87
ID	95.52	**97.22**	94.29	95.52	**98.97**	97.50	97.57	98.45
O	**99.92**	99.90	99.89	99.89	**99.95**	99.93	99.93	99.90

This table contains the performances of different medical models compared to the baseline model (*RoBERTa*) on 2 different medical datasets.
The 3 different medical models are:

- **PubMedBERT (pm)**[19]: This *BioMedBERT* model is pretrained from scratch using abstracts from PubMed and achieves the state-of-the-art performance on several biomedical NLP tasks.

```
model =
    AutoModelForTokenClassification("microsoft/BiomedNLP-BiomedBERT-base-uncased-
 abstract",num_labels=num_labels)
```

- **SciBERT (sci)**[6]: This *BERT* model is pretrained from scratch using the complete papers coming from *Semantic Scholar* that represents a corpus size of 1.14M of papers.

```
model =
    AutoModelForTokenClassification("scibert_scivocab_uncased",num_labels=num_labels)
```

- **BioBERT (bio)**[28]: This *BERT* model is pre-trained from scratch using biomedical articles from PubMed and PMC.

```
model =
    AutoModelForTokenClassification("dmis-lab/biobert-base-cased-v1.2",
num_labels=num_labels)
```

Table 4.8: Performance Comparison of variants of *BERT* model

Label	**i2b2 2006**				**i2b2 2014**			
	BC	*BU*	*LC*	*LU*	*BC*	*BU*	*LC*	*LU*
PERSON	92.33	95.04	93.56	**96.06**	96.87	98.84	96.81	**98.94**
LOCATION	88.83	92.41	91.41	**92.87**	94.81	96.44	95.52	**97.27**
DATE	97.85	99.17	98.27	**99.30**	96.92	98.91	96.78	**99.14**
ID	95.24	**98.59**	95.08	91.43	95.63	**99.08**	95.58	98.37
O	99.78	99.92	99.81	**99.92**	99.82	99.93	99.83	**99.94**

BC	*google-bert/bert-base-cased*
BU	*google-bert/bert-base-uncased*
LC	*google-bert/bert-large-cased*
LU	*google-bert/bert-large-uncased*

Base models are resource-efficient, while Large models offer better performance with higher computational needs. Cased models retain text casing, useful for case-sensitive languages, whereas Uncased models ignore casing, suited for case-insensitive contexts.

Table 4.9: Performance of different models on i2b2 2014

Label	*RoBERTa*	*XLM-RoBERTa*	*mBERT*	*BERT*	*DistilBERT*	*AlBERT*
DOCTOR	98.38	**98.99**	97.65	98.13	96.30	97.30
PATIENT	**99.02**	98.85	98.35	98.83	96.44	97.73
DATE	99.31	**99.48**	98.97	99.08	98.85	98.71
COUNTRY	**89.04**	83.58	85.52	85.14	82.76	81.63
CITY	**93.06**	92.84	90.32	92.65	89.91	87.84
STREET	98.82	**99.40**	97.35	96.78	95.36	97.13
ZIP	100.00	100.0	100.00	100.0	99.11	99.55
STATE	**98.36**	94.81	93.20	96.05	92.31	87.87
HOSPITAL	95.66	**96.33**	95.34	95.14	94.04	94.91
MEDRECORD	**99.37**	98.87	99.05	98.89	98.42	98.73
PHONE	97.94	98.77	98.76	98.77	96.69	97.52
O	99.94	99.94	99.91	99.92	99.90	99.91

For the new table, models that have not been previously discussed are introduced:

- **XLM-Roberta** [11]: This *XLM-Roberta* model is a scaled-up version of *BERT* that supports multilingual text processing, pre-trained on 2.5TB of filtered CommonCrawl data in 100 languages.

```
model = AutoModelForTokenClassification("xlm-roberta-large",
    num_labels=num_labels)
```

- **BERT** [15]: This foundational *BERT* (Bidirectional Encoder Representations from Transformers) model is pre-trained using a large corpus of unlabelled text including the entire Wikipedia (in English) and Book Corpus.

```
model = AutoModelForTokenClassification("bert-large-uncased",
    num_labels=num_labels)
```

- **DistilBERT** [39]: This *DistilBERT* model is a smaller, faster, cheaper, and lighter version of *BERT*, distilled using the teacher-student paradigm but retaining 97% of its predecessor's language understanding capabilities.

```
model = AutoModelForTokenClassification("distilbert-base-uncased",
    num_labels=num_labels)
```

- **ALBERT** [26]: *ALBERT* (A Lite *BERT*) is an optimized version of *BERT* that achieves state-of-the-art results on various NLP benchmarks while using far fewer parameters than standard *BERT* models by factorizing the embedding layer and sharing parameters across the hidden layers.

  ```
  model = AutoModelForTokenClassification("albert/albert-base-v2",
      num_labels=num_labels)
  ```

This result, along with similar experiments on the i2b2 2006 dataset A.6.3 and MEDDOCAN A.6.4, clearly shows that three models outperform the others: *RoBERTa*, *XLM-RoBERTa*, and *mBERT*.

4.2.5 Parameters Performances

This section explores the impact of three crucial parameters on the performance of models for de-identifying clinical documents: batch size, number of training epochs, and learning rate. The objective is to evaluate each parameter's influence and establish the most effective settings for enhancing accuracy and efficiency in clinical text processing.

Batch Size Results

Table 4.10: F_1 Scores for Different Batch Size Across Various Configurations i2b2 2014

Label	*1*	*2*	*4*	*8*	*16*	*32*
DOCTOR	98.04	98.16	**98.70**	98.20	98.63	98.27
PATIENT	98.74	98.74	**99.16**	98.97	99.15	98.88
DATE	99.35	99.24	99.36	**99.43**	99.39	99.41
COUNTRY	83.45	**92.00**	88.44	88.89	89.04	89.04
CITY	91.63	91.73	92.65	**94.78**	94.12	92.98
STREET	97.35	98.22	**98.82**	98.06	98.37	99.25
ZIP	100.00	100.00	99.12	99.56	99.56	100.00
STATE	96.08	**98.03**	96.64	97.40	97.33	97.03
HOSPITAL	96.22	96.50	97.02	**97.30**	97.14	96.19
PHONE	98.77	97.12	**98.78**	98.35	97.94	97.94
O	99.94	99.95	99.95	99.95	99.95	99.95

With this result and the result on i2b2 2006 in Appendix A.6.5, it can be seen that a good batch size is between 2 and 8, with the optimal batch size being 4.

Epochs Results

Table 4.11: F_1 Scores for Different Epochs Across Various Configurations i2b2 2014

Label	*1*	*2*	*5*	*10*	*20*
DOCTOR	97.91	98.13	**98.70**	98.55	98.51
PATIENT	98.92	99.16	99.16	99.30	**99.48**
DATE	99.32	99.20	99.36	**99.39**	99.28
COUNTRY	87.01	88.74	88.44	**94.12**	93.42
CITY	94.83	95.45	92.65	93.91	**97.89**
STREET	98.37	98.22	**98.82**	98.22	98.52
ZIP	100.00	99.56	99.12	100.00	100.00
STATE	94.67	96.79	96.64	96.99	**97.11**
HOSPITAL	96.19	96.77	97.02	96.93	97.02
PHONE	96.75	97.52	**98.78**	97.50	97.52
O	99.94	99.94	99.95	99.95	99.95

It can be seen from this table that the best performances are achieved between 5 and 20 epochs, but a trade-off must be considered to avoid overfitting.

Learning Rate Results

Table 4.12: Performance metrics for different entities across various learning rates on *i2b2 2014* dataset

ID	**5e-5**	**3e-5**	**2e-5**	**1e-5**	**5e-6**
DOCTOR	98.36	98.47	98.59	**98.63**	98.08
PATIENT	99.06	99.02	**99.30**	98.74	98.50
DATE	99.36	99.41	99.41	99.43	99.43
COUNTRY	88.89	**91.16**	88.89	87.67	86.49
CITY	92.86	94.36	93.03	94.14	**94.59**
STREET	98.37	**98.81**	97.91	97.92	97.60
ZIP	99.56	99.56	100.00	100.00	100.00
STATE	95.36	**97.07**	97.03	96.73	97.05
HOSPITAL	97.04	**97.14**	96.96	96.89	96.01
PHONE	98.35	98.35	97.54	98.35	97.94
O	99.95	**99.96**	99.95	99.95	99.94

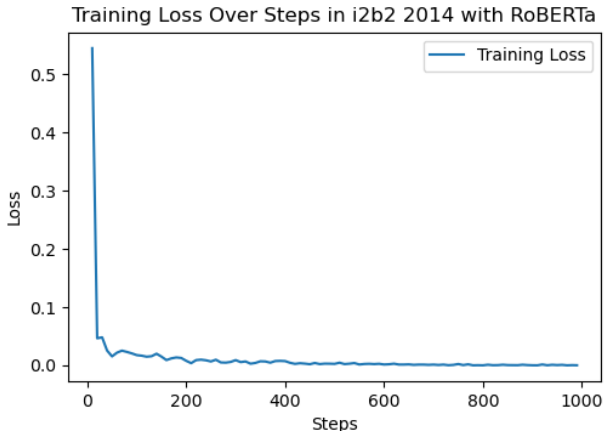


Figure 4.3: Loss Function with learning rate of $5e^{-5}$

The same experiment has been done with the *MEDDOCAN* dataset in Appendix A.6.6, and the results demonstrate that the best learning rate is between $5e^{-5}$ and $3e^{-5}$.

4.3 Discussion of Traditional Approaches

4.3.1 Evaluation of BIO-Schema Removal

The BIO Schema traditionally assists in identifying the boundaries of annotated entities within text, which is critical for precisely delineating the start and end points of entities. This study tests the hypothesis that removing the BIO-schema might improve performance.

Table 4.2 compares the F_1-scores with and without BIO encoding across various labels on the i2b2 2014 dataset. The results indicate improvements or consistent performance even after removing the BIO markup. For example, the F_1-score for the DOCTOR label improved from 97.9% to 98.4% upon removal of BIO encoding. These results, consolidated with experiments in other dataset such as *Meddocan* suggest that models can effectively infer boundaries without explicit BIO markup, potentially simplifying the training process and reducing the model's dependency on specific annotation schemes. These results are confirmed with the study of Tehseen et al. [46] that complete on their setup a F_1-score of 83.75 without BIO-Tag instead of 75.55 with BIO-Tag.

Given the token-specific approach, where the model's accuracy is measured by its ability to classify each token correctly regardless of its position within an entity, the positive results from removing the BIO-schema align well.

4.3.2 Effectiveness of Group Labeling

Upon reviewing the limitations posed by data scarcity and inconsistent labeling across datasets, this study introduced a grouped labeling strategy to simplify label management and enhance model transferability. This strategy involves consolidating detailed labels into broader categories such as PERSON, LOCATION, DATE, and ID, as depicted in Table 4.4. These categories are designed to encapsulate all possible labels across datasets, facilitating the application of transfer learning techniques and respecting the HIPAA definition of PHI.

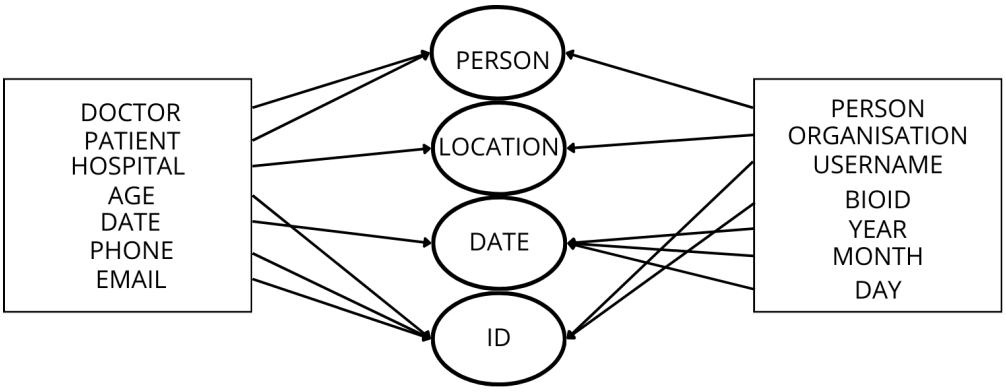


Figure 4.4: Mapping strategy for label generalization across datasets.

The use of grouped labels has demonstrated clear performance benefits. Table 4.3 shows that the F_1-scores for grouped labels are consistently higher or equal compared to scores using detailed labels. For instance, the PERSON category, which aggregates several individual labels, shows an improved F_1-score from 98.7% to 99.3%.

The benefits of this grouping strategy not only simplify model training and annotation efforts but also enhance the model's adaptability and scalability across diverse datasets. Although some specific information might be lost due to label consolidation.

4.3.3 Language Model Performance Analysis

A significant challenge encountered in this study is overcoming language barriers. The evaluation of different language-specific and multilingual models across various datasets reveals several important insights:

1. **Language-Specific Models' Superiority:** Models trained on specific languages generally outperform others on their native datasets, supporting the idea that language-specific training yields better results.

2. **Multilingual Models' Versatility:** Multilingual models, like *mBERT*, show strong performance across different languages, highlighting their robustness and capability to handle diverse linguistic nuances.

3. **RoBERTa's Cross-Linguistic Potential:** Despite being trained only on English, *RoBERTa* demonstrates good performance on datasets in other languages, indicating its potential for cross-linguistic applications.

However, this study also demonstrates a "case by case" situation where no single model universally outperforms others. The effectiveness of a model depends heavily on the specific dataset and language in question. While language-specific models excel in their native contexts,

multilingual models provide promising results across various languages, and *RoBERTa* shows surprising adaptability.

Overall, these findings underscore the complexity of model adaptability in different linguistic environments and **the need for tailored approaches depending on the dataset**.

4.3.4 Impact of the Model

In the continued effort to optimize de-identification processes, a critical step is selecting the most effective model.

Evaluation of General vs. Medical-Specific Models

Initial tests focused on determining the suitability of models specialized in medical documents compared to general models. Surprisingly, the results, as depicted in Table 4.7, indicate that general models, particularly *RoBERTa*, are better suited. This may be attributed to the nature of the labels involved in medical documents such as 'PERSON' or 'LOCATION,' which are generic rather than strictly medical terms. These findings suggest that general models, with their broad training on diverse data, are more adaptable to the generalized labels commonly found in medical datasets.

Variations in Model Architecture

Further analysis involved comparing different model architectures: *BASE, LARGE, CASED*, and *UNCASED*. Table 4.8 presents these findings. The *LARGE UNCASED* variant of the models consistently outperformed the others. This result was somewhat expected, as larger models generally have a greater capacity to learn and generalize from data. Additionally, using an uncased model removes the complications of handling uppercase and lowercase distinctions, making the learning process simpler and more consistent.

Comparative Analysis of Top Performing Models

Having established a preference for *LARGE UNCASED* and general models, specific models fitting this description were evaluated across various datasets. The comparative results across i2b2 2014, i2b2 2006, and Meddocan datasets are detailed in Table 4.9, and in Appendix A.6.3 and A.6.4.
RoBERTa and *XLM-RoBERTa* emerged as top performers, demonstrating excellent adaptability and robustness across datasets. Following closely by *mBERT* and *BERT*, which also showed good performance. As anticipated, smaller or more specialized models like *DistilBERT* and *AlBERT* did not perform as well, likely due to their reduced model complexity and focus.

4.3.5 Optimization of Model Parameters

The final aspect of this study involves a detailed examination of model parameter tuning, specifically exploring the influence of batch size, number of epochs, and learning rate on the performance of de-identification models.

Batch Size Batch size significantly affects model performance. For the i2b2 2014 dataset, a batch size of 4 yields the best results, whereas for the 2006 dataset, a smaller batch size of 2 is preferable. To achieve balanced performance, a batch size of 4 will be predominantly used in subsequent phases of the research.

Epochs Size The number of epochs, which refers to the number of iterations over the entire dataset during training, is another pivotal parameter. Evaluation indicated that performance was good with epochs ranging from 5 to 20. To balance time resources and **avoid overfitting**, the decision was made to continue working with 5 epochs.

Learning Rate The learning rate determines the step size at each iteration while moving toward a minimum of a loss function. For the i2b2 2014 dataset, a learning rate of $3e^{-5}$ provides optimal results, while for the Meddocan dataset, a slightly higher rate of $5e^{-5}$ is more effective. To maintain consistency with the standard practice, a learning rate of $5e^{-5}$ is used.

4.3.6 Conclusion

This research has explored the foundational aspects of de-identification in clinical documents using deep learning. Through rigorous evaluations, the study has established a comprehensive understanding of the strengths and limitations of various deep learning models, including *RoBERTa* and *mBERT*, across different datasets and languages in single dataset focus. This exploration has underscored the potential of these models to adapt to diverse linguistic contexts, enhancing their utility in global healthcare applications.

Key findings from the study emphasize the effectiveness of grouped labeling strategies in managing data scarcity and label inconsistencies. This approach not only simplifies the training process but also enhances the transferability of models, thereby facilitating more efficient computational implementations.

Moreover, the research has provided valuable insights into the optimal configurations of model parameters such as batch size, number of epochs, and learning rate, which are critical for maximizing the performance of de-identification systems without risking overfitting.

At present, the research has primarily focused on the approach discussed in the problem statement: working within the same dataset. Thanks to these results, excellent outcomes have been achieved. However, the necessity for a high-quality, gold standard annotated dataset remains a significant constraint.

With the foundational knowledge and empirical data gathered from this phase, the study is now poised to advance into the next phase of research: cross-dataset application through transfer learning. This progression will allow us to rigorously compare the performance of models trained on one dataset and applied to another.

4.4 Evaluation of Transfer Learning

Having evaluated models in a straightforward setup, this section will now explore the performance with transfer learning. The aim is to determine whether transfer learning can yield good results or if its associated challenges are too significant, necessitating a continued focus on a "single dataset" approach.

4.4.1 Naive Transfer Learning

Initially, it is crucial to establish the baseline performance of the models without any fine-tuning. To this end, the two most promising models, *RoBERTa* and *mBERT*, were employed. These models were evaluated across different medical datasets to understand the impact of naive transfer learning.

Results

Table 4.13: Transfer Learning Performance without Fine-Tuning

Label	**i2b2 2014 to Meddocan**		**i2b2 2006 to i2b2 2014**		**i2b2 2014 to i2b2 2006**	
	Roberta	*mBERT*	*Roberta*	*mBERT*	*Roberta*	*mBERT*
PERSON	81.91	**91.21**	**95.47**	88.29	56.40	**65.79**
LOCATION	63.20	**63.50**	**84.97**	72.65	78.00	**78.85**
DATE	**95.76**	93.26	**83.38**	82.79	80.65	**83.26**
ID	66.43	**67.40**	**67.27**	52.19	92.10	**95.66**
O	98.63	**98.68**	**99.67**	99.54	99.27	**99.32**

The model's performance on the respective testing sets was promptly assessed. This approach guarantees uniformity in evaluation procedures across different datasets. The table demonstrates the challenges of applying transfer learning without fine-tuning across three distinct transfer scenarios. The analysis reveals poor performance, with significant inconsistencies and limited effectiveness across various labels and datasets. These findings clearly indicate the critical need for fine-tuning to achieve acceptable performance levels when deploying models across different medical environments.

4.4.2 Impact of the Fine-Tuning Process

This analysis focuses on assessing the impact of different training strategies for deep learning models applied to the task of de-identifying clinical documents. Three primary strategies are considered:

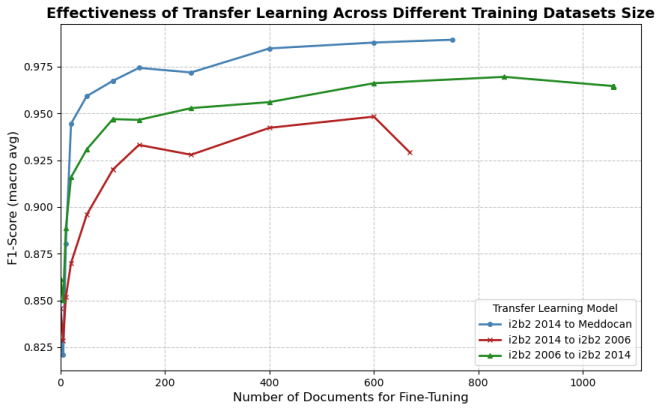
1. **Training from Scratch (scratch):** Models pre-trained are fine-tuned exclusively on the target dataset from the beginning, without any transfer learning.
2. **Naive Transfer Learning (naive):** Models that have been fine-tuned on a specific dataset are directly applied to the target dataset without any additional fine-tuning. This strategy evaluates how well the model performs on unseen dataset.
3. **Fine-Tuning:** Models that have already been fine-tuned on a specific task are further trained on the complete target dataset's training set to better adapt to the specific task. The test set is then used for evaluating the performance.

Table 4.14: Comparison of different Transfer Learning methods

Label	**i2b2 2014 to Meddocan**			**i2b2 2006 to i2b2 2014**			**i2b2 2014 to i2b2 2006**		
	Scratch	*Naive*	*Fine-Tune*	*Scratch*	*Naive*	*Fine-Tune*	*Scratch*	*Naive*	*Fine-Tune*
PERSON	**99.88**	91.21	99.56	**99.34**	95.47	97.87	**95.96**	65.79	88.50
LOCATION	**99.29**	63.50	97.78	**97.85**	84.97	93.65	**94.19**	78.85	87.88
DATE	**99.69**	93.25	99.33	**99.26**	83.38	97.61	**99.30**	83.26	96.48
ID	**99.15**	67.40	98.15	**98.97**	67.27	93.32	**97.06**	95.66	91.86
O	**99.95**	98.67	99.89	**99.95**	99.67	99.87	**99.87**	99.32	99.84

Before advancing, we assessed transfer learning performance across varying training dataset sizes, rather than using the full set as in earlier Fine-Tune assessments.

Figure 4.5 illustrates how transfer learning effectiveness changes with the training volume, aiming to identify the minimal document count required for optimal performance.



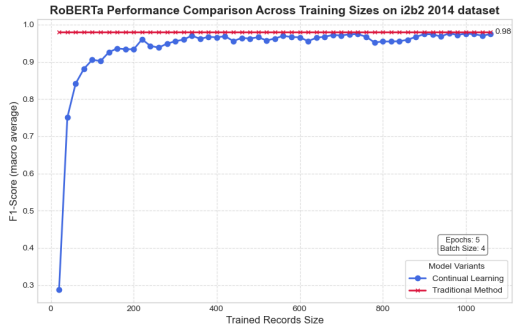
Models are fine-tuned on specific datasets and then incrementally fine-tuned with increasing numbers of new documents, specifically 0, 5, 10, 20, 50, 100, 150, and so on. Evaluations are conducted on test set to assess the performance enhancements achieved through this transfer learning approach.

Figure 4.5: Effectiveness of transfer learning models with varying fine-tuning dataset sizes.

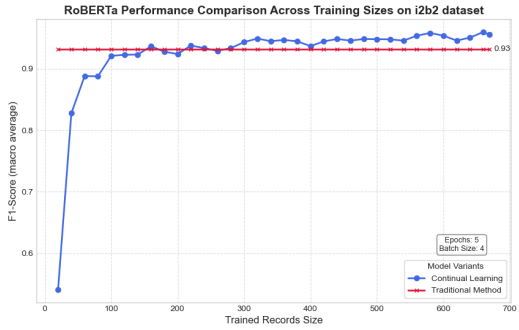
For the 2 results, *mBERT* is used for *i2b2 2014 to Medocan* and *i2b2 2014 to i2b2 2006* while *RoBERTa* is used for *i2b2 2006 to i2b2 2014*.

4.4.3 Speed of Training without Transfer Learning

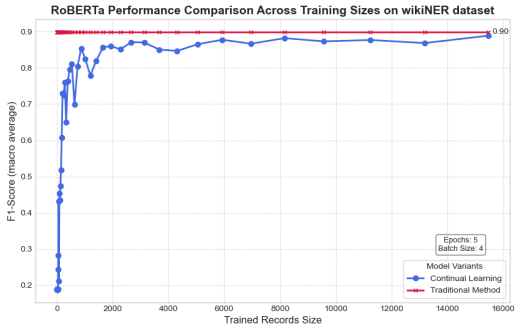
This section evaluates the F_1-score of models trained from scratch on various datasets including i2b2 2006, i2b2 2014, and wikiNER. It discusses how model performance changes with the training size, highlighting that deep learning models achieve satisfactory results with few documents, without the need for transfer learning.



(a) i2b2 2014 Dataset



(b) i2b2 2006 Dataset



(c) wikiNER Dataset

Figure 4.6: F_1-score performance depending on the number of training documents used.

4.5 Discussion of Transfer Learning

4.5.1 Effectiveness of Naive Transfer Learning (without Fine-Tuning)

Evaluating naive transfer learning, where models trained on one dataset are applied to a different target dataset without fine-tuning, shows significant performance degradation. This approach struggles due to language differences, label inconsistencies, and unstructured data formats.

Table 4.13 highlights the poor results. For example, in the PERSON label category, F_1-scores were low: *RoBERTa* scored 56.40 and *mBERT* scored 65.79 when transferring from i2b2 2014 to i2b2 2006. Similarly, transferring from i2b2 2014 to Meddocan yielded F_1-scores of 63.50 and 67.40 for the LOCATION and ID labels, respectively. These results show that naive transfer learning does not effectively capture target data specifics.

Unfortunately, the effectiveness of naive transfer learning is so limited that it underscores the necessity of continuing labor-intensive manual annotations on the target dataset to obtain enough annotated documents to fine-tune the model on this target dataset.

Overall, the poor performance emphasizes the need for advanced techniques and fine-tuning for better results in clinical document de-identification. Studies have shown that while transfer learning offers a foundational knowledge base, fine-tuning is essential for adapting to specific target data, underlining the limitations of naive transfer learning [21].

But the table 4.14 reveals that even when employing the complete training set for fine-tuning, models do not consistently outperform those trained from scratch. This suggests that models perform optimally only on familiar datasets, and applying transfer learning can introduce noise rather than value. These results demonstrate that transfer learning is not a viable option in fields where perfect de-identification is required.

Impact of Training Set Size on Model Performance

The investigation has reached a crucial juncture. A primary challenge identified in this study is the scarcity of adequately annotated datasets, which is a significant barrier to advancing model training and validation. This shortage leads us to a critical question:

How many documents are truly necessary to achieve satisfactory performance?

In response, a new avenue was explored, quantifying the minimum number of documents required for effective model training. This inquiry is particularly relevant as it aims to determine if hospitals can achieve high-performance de-identification with a smaller, more manageable set of annotated documents, thereby reducing the cost and effort associated with extensive manual annotation.

The findings, depicted in Figure 4.5, reveal that transfer learning models require surprisingly few additional training documents to register significant performance enhancements. This discovery is pivotal as it indicates that even a limited but relevant set of training data can substantially improve the effectiveness of models employing transfer learning strategies. This outcome provides a hopeful perspective that efficient and cost-effective model training is achievable with fewer resources than previously anticipated.

Advantages of Training from Scratch on Small Datasets

The experiments, as depicted in Figures 4.6a, 4.6b, and 4.6c, have provided critical insights into the effectiveness of models trained from scratch with few documents. For instance, the i2b2 2014 dataset achieves performance levels close to its maximum with only 220 documents. These models not only learn quickly but also reach high levels of performance with a relatively small number of training documents.

4.5.2 Future Directions

The results underscore the potential to significantly reduce the volume of data required for effective model training. Building on this, one promising direction for future research is the creation of a specialized tool that leverages the rapid learning capabilities of models trained from scratch. This tool would aim to facilitate and accelerate manual annotations, allowing for the rapid training and deployment of effective models with minimal data. Such a development could address one of the critical challenges identified in the problem statement: the extensive effort and resources currently required for manual data annotation.

4.6 Conclusion

This study has highlighted the poor performance of transfer learning in certain scenarios, emphasizing the necessity of staying in single dataset focus using only few documents to achieve good results. The findings suggest that models trained on small, specific datasets can perform well quickly and efficiently, underscoring the limitations of transfer learning in clinical NLP applications.

As a direct response to these findings, the development of a new annotation tool is proposed. This tool will utilize the principles observed in the experiments to expedite the manual annotation process, thereby enabling the efficient creation of highly accurate models with reduced dataset requirements. This approach not only aligns with the discovery but also offers a practical solution to the challenges posed by the need for large annotated datasets in clinical environments.

Chapter 5

Development and Evaluation of the Incremental Learning Annotator

Practical Informations

All the open-source code of ILA tool can be found on Github at this URL: `https://github.com/NicolasHuberty/DeepDeId-ClinicalDocs/tree/main`
A demonstration of the tool is also available on Youtube from this link: `https://youtu.be/PJTIBT_-VHk` or directly with the QR-code.

5.0.1 How the Tool Addresses the Problem Statement

The tool addressed in this chapter overcomes several critical challenges faced by hospitals in developing effective Natural Language Processing (NLP) models for clinical document de-identification. These challenges include managing unstructured and diverse data formats, language diversity, and inconsistencies in labeling. By focusing on a single hospital dataset, the tool simplifies and enhances the annotation process, thereby significantly reducing the time and costs associated with manual annotations.

Previous studies, such as those by Grouin et Névéol [18] and South et al. [40], have explored rule-based and machine learning pre-annotation methods, yet these have often resulted in high rates of inaccuracies in pre-annotation. ILA introduces a novel application of deep learning to overcome these limitations, offering a more adaptable, efficient, and open-source solution for the pre-annotation of clinical documents.

Key Benefits:

- **Specialization in Data Formats:** The tool's exclusive focus on a specific dataset allows it to handle diverse and unstructured data more accurately by becoming highly specialized in that dataset's particular structures.

- **Handling Language Uniformity:** By limiting its scope to data from a single linguistic background, the tool avoids the complexities of multilingual data processing, thereby enhancing predictive accuracy.
- **Label Customization:** The autonomy in labeling allows the tool to adapt to the hospital's specific needs without the constraints of external labeling standards, improving the precision of data annotation.

Economic and Efficiency Advantages: The development of high-quality annotated datasets is both costly and labor-intensive. This tool, named the Incremental Learning Annotator (ILA), mitigates these issues by requiring fewer documents for effective training and by speeding up the annotation process. The approach not only curtails direct costs but also aligns with the hospital's operational needs by facilitating the development of robust, high-performing NLP systems specifically tailored to its dataset.

5.0.2 General Concept of the Incremental Learning Annotator

ILA represents a transformative approach to NER within clinical environments, merging state-of-the-art deep learning technologies with incremental learning principles to provide a dynamic, adaptive solution. This tool is specifically designed to address the need of large, well-annotated documents to train an effective model.

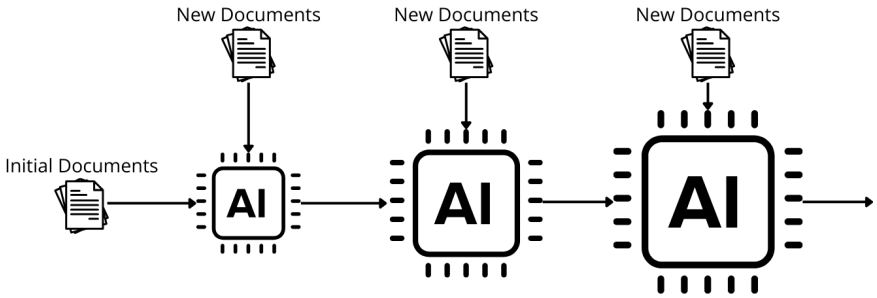


Figure 5.1: Schematic representation of the Incremental Learning Process in ILA.

Core Features of the Incremental Learning Annotator

- **Dynamic Learning and Adaptation:** ILA is engineered to continuously refine its weights through incremental learning in integrating new data as they become available, without relying on extensive pre-annotated datasets. This feature reduces dependence on large annotated dataset to obtain a good model.
- **No need for annotated documents**: Unlike some alternatives, this tool can be used without any pre-annotated document. There is no need to configurate a model with annotated data. Everything is done automatically on this tool.

- **Efficiency in Annotation:** By using techniques such as prioritizing less confident documents first, the model learns quickly and improves its performance over time.
- **Real-Time Performance Evaluation:** The tool evaluates model performance in real time, providing immediate feedback on effectiveness. With real-time evaluation, the institution can determine when it is no longer necessary to continue manual annotation to label the complete dataset with the actual model performance.

Operational Modes

ILA supports two primary operational modes, addressing diverse requirements in clinical data handling:

1. **Enhanced Semi-Manual Annotation:** In this mode, ILA aids semi-manual annotation processes, providing a high-performance model that significantly expedites annotation tasks while ensuring data accuracy. Ideal for scenarios requiring detailed manual review, such as clinical de-identification, this mode combines manual precision with the automated efficiency of state-of-the-art deep learning models.
2. **Automated Model Training:** This mode enables organizations to define specific accuracy thresholds that suit their operational needs. During the annotation process, human-annotated documents are dynamically divided into training and evaluation sets. Training continues until the model achieves or surpasses the predefined accuracy threshold, after which the model can be used to annotate autonomously new documents. This ensures the model's reliability and accuracy before full-scale deployment.

These functionalities make ILA an invaluable asset for any organization looking to enhance the efficiency and effectiveness of their NER applications, particularly in the demanding and dynamic field of healthcare.

Comparison with Existing Annotation Tools

While many existing tools offer interfaces for manual annotation, ILA differentiates itself in several key areas. Here, the tool is compared with Prodigy, a widely recognized annotation tool, and other similar tools such as Label Studio:

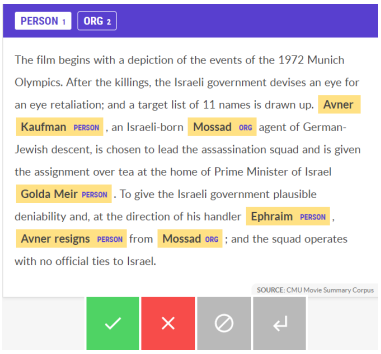


Figure 5.2: Demo of Prodigy Interface

- **Accessibility and Open Source:** This tool is open-source, user-friendly, requiring no programming skills, unlike Prodigy which demands coding for deep learning model integration.
- **Flexibility in Model Training:** It accepts untrained models, unlike Prodigy, which typically requires an existing model that has already been fine-tuned on the target dataset.
- **Incremental Learning:** Each new document processed enhances the tool's learning capability, unlike Prodigy which does not inherently support incremental learning.
- **Adaptive Performance Metrics:** Real-time monitoring allows users to assess whether the model's accuracy meets their specific requirements.

These features position ILA as a superior alternative in environments where rapid, reliable, and straightforward solutions are essential.

5.1 Operational Methodology

5.1.1 General Architecture

First Step: Configuration of ILA

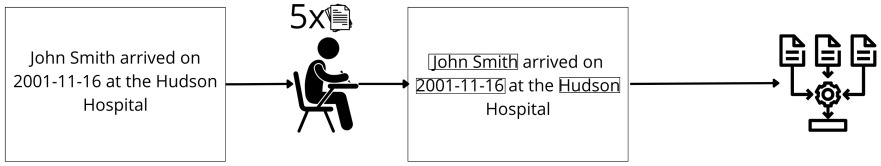


Figure 5.3: First step for the Incremental Learning Annotator

Setting Up the Model: To initiate the model, the user is prompted to create a project encompassing all relevant documents. Crucially, the **first five documents**[1] are manually annotated by a human annotator. This manual annotation phase is vital for establishing the linear layer of the neural network, enabling it to begin making predictions for subsequent documents.

[1]The design of the project is highly modular, allowing for significant adaptability in its setup. Users can alter the initial number of documents required for setup with a simple parameter change. Although adjustable, five documents have been identified as the optimal number for initial training due to considerations of processing speed and immediate model effectiveness.

Second Step: Refining Model Predictions

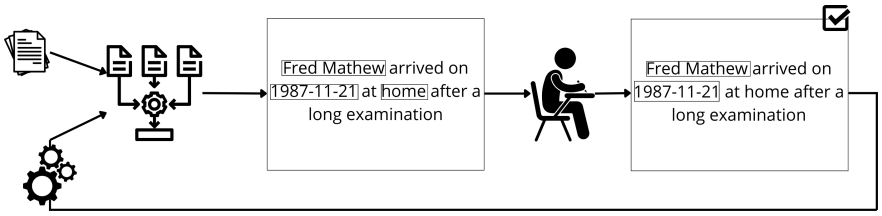


Figure 5.4: Second step for the Incremental Learning Annotator

In the second step of ILA process, the model's predictive performance is refined through active user intervention. This phase involves users playing a critical role in correcting the model's initial predictions, which significantly boosts the model's accuracy early in its training phase.

During this phase, all documents are utilized for training purposes to maximize learning opportunities. This iterative correction and learning process ensures rapid improvement in performance, setting a strong foundation for subsequent phases of deployment and broader usage.

Third Step: Evaluation of the Model

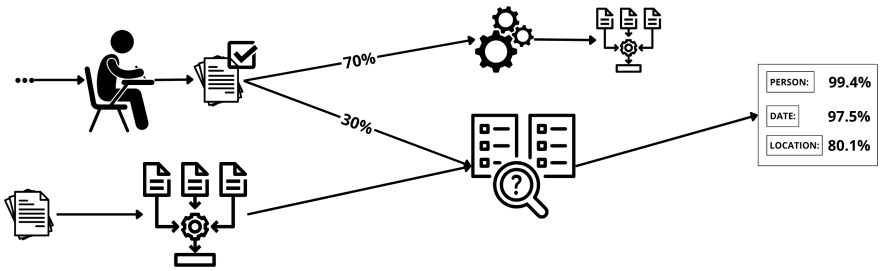


Figure 5.5: Third step for the Incremental Learning Annotator

The third step in ILA process marks a significant transition in the model's training and evaluation methodology. At this stage, having been refined through iterative user corrections, the model begins to show robust predictive capabilities. This phase introduces a strategic shift from solely training to a balanced approach that incorporates both training and evaluation.

Introduction of the Evaluation Set: When the model received a determined number of documents annotated, the **evaluation percentage** parameter is introduced. This parameter determines the proportion of new incoming documents that are reserved for the evaluation set. This parameter is defined at the creation of the project and can be modified in real-time like all other parameters. This set serves to test the model's effectiveness, providing a clear benchmark for its performance.

This balanced approach ensures that while the model continues to learn from ongoing training data, it also undergoes rigorous testing to assess its real-world applicability and accuracy.

5.2 Technical Workflow

This will cover how the tool handles incremental learning cycles and integrates with existing data systems. This section is divided into three parts:

5.2.1 Creation of the Project

Supported Data Formats

The tool supports a wide range of data formats to accommodate various sources of clinical and textual data. Supported formats include TSV and CSV, where records are separated by empty lines; ConLL-U format, widely used in NLP tasks; BRAT format, commonly used for annotating text, especially in biomedical contexts; and plain TXT files, which must use a specific marker (<RECORD_SEPARATION>) to delineate separate records.

Each format requires a specific structure to ensure that the tool correctly parses and processes the data for annotation and learning.

Data Annotation and Label Management The database handles two scenarios for label management:

1. If the dataset is already annotated, it uses the real labels to simulate annotations and verify the tool's performance.

2. For unannotated datasets, all manual labels are set to 'O', indicating the absence of labels.

Database and Data Handling

Upon project creation, a *SQLite* database is initialized to manage the data efficiently. The database stores records in a randomized order to prevent the model from learning any unintended patterns or chronological changes in data structure, which could bias the learning process. This approach ensures the model's robustness and adaptability to varying data formats and structures over time. The database includes several columns that are described in Appendix A.7.1.

Configuration and Setup Files

Project creation triggers the generation of another critical file: A *JSON* configuration file detailing the setup and parameters of the project. This file includes paths to the saved model and tokenizer, label definitions, parameters of the project and UI color settings for labels. The example of this configuration file can be found in Appendix A.7.2.

These mechanisms ensure that the tool is not only functional but also efficient in managing data throughout its lifecycle, from initial annotation through model training and evaluation.

5.2.2 Manager: Orchestrating Model Operations

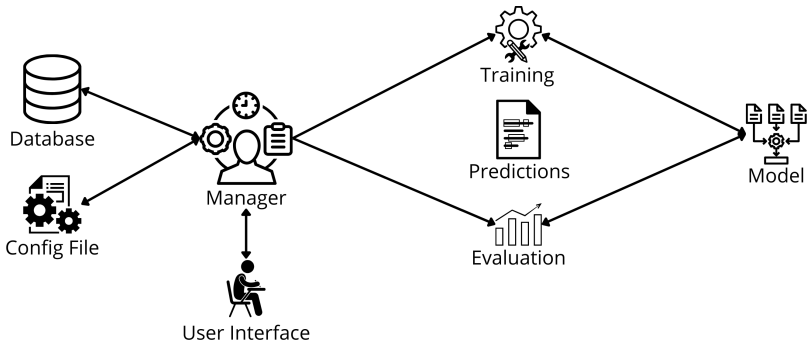


Figure 5.6: Diagram illustrating the Manager's central role as an interface between user interactions, data processing, and model operations within ILA.

The Manager acts as a central orchestrator within ILA, effectively handling API calls and executing Python functions to manage the flow of data and model operations.

Operational Flow

The core function within the Manager, `handle_new_record`, is tasked with loading documents and determining their allocation to training or evaluation sets based on specific criteria, such as document freshness or predefined thresholds.

Function Execution: Withing the execution of `handle_new_record`, the function `process_new_record` is called, which orchestrates:

1. **The training of the model:** The model undergoes training with the newly available data, enhancing its learning based on recent inputs if the determined threshold is reached.
2. **Predict new documents:** Following training, the model performs predictions on the records to immediately gauge the impact of its latest learning.
3. **Launch the Evaluation:** Depending on the availability of data, the model's performance is evaluated through two primary scenarios:
    - If a user-provided annotated testing set, the model leverages this dataset to predict and evaluate its performance, offering a direct measure of its efficacy.
    - In the absence of a user-provided testing set, documents designated for evaluation within the operational flow are used to test the model's effectiveness.

The manager plays a central role between the user interface and all implementations, but it is not crucial to pass by him, all implementation parts (training, prediction and evaluation) can be called and used without being launched by the manager.

5.2.3 Training the Model

Data Loading and Preparation

Before training, the system loads the necessary records from the database, checking if they meet the criteria for training (e.g., they are not already used for training). It then loads the model and tokenizer from specified paths or initializes new ones if none exist. The data are then tokenized and converted into a format suitable for training:

Model and Tokenizer Management

When creating a project, the user selects the model that will be used. If the model has already been trained, the training phase will retrieve the model and tokenizer with the same settings as when it was last used, including the optimizer, weights, and learning rate.

```
save_model_optimizer_scheduler(model,
    optimizer, scheduler, model_path)
tokenizer.save_pretrained(tokenizer_path)
save_config_field(project_name,
    "model_path", model_save_path)
save_config_field(project_name,
    "tokenizer_path", tokenize_path)
```

Upon completion of training, the system saves the updated model and tokenizer to new paths, ensuring the latest state is preserved. This includes handling file directories and updating configuration fields to reflect the new paths.

Training Loop

```
for epoch in range(5):
    for batch in train_dataloader:
        optimizer.zero_grad()
        outputs = model(**inputs)
        loss = outputs.loss
        loss.backward()
        optimizer.step()
        scheduler.step()
```

The training loop runs for a predetermined number of epochs. During each epoch, the model processes batches of data, calculates loss, and updates its weights accordingly. After each batch, the optimizer adjusts the learning rate, and the scheduler updates the training schedule based on the progress.

Updating Training Metadata

Finally, the system updates the project configuration to reflect the total number of records trained and adjusts the training queue accordingly. This ensures that the project status accurately reflects the current state of the model:

```
    save_config_field(project_name, "totalTrain", total_train_records +
len(ids))
    save_config_field(project_name, "numRecordsToTrain", num_records_untrain
- len(ids))
```

5.2.4 Prediction and Alignment

Aligning the predicted labels accurately with the original text represents a significant challenge in NLP, especially when it needs to adapt to various models and tokenizers.

Challenges in Alignment

Tokenizers split text into tokens in a manner that does not correspond directly to the original word boundaries. Additionally, different tokenizers may produce varying sequences of tokens for the same text, further complicating the alignment of prediction outputs back to the original text.

Strategic Approach to Alignment

The alignment function addresses these challenges by mapping each token's prediction back to its corresponding word in the original text. The function performs the following steps:

1. **Tokenization:** The text is broken down into smaller tokens, using a sliding window approach with overlap. This ensures comprehensive coverage and proper handling of all words, even in longer texts, by maintaining context at chunk boundaries.

2. **Prediction and Confidence Calculation:** For each tokenized chunk, predictions are made, and confidence scores are calculated. The softmax function helps determine the most likely label for each token by providing a normalized probability score.

3. **Mapping Predictions to Original Words:** During tokenization, if a word is split into multiple tokens, only the first token's prediction is evaluated, while the subsequent tokens are set to -100 to be ignored by the model. The function aggregates these predictions, ensuring that each word is represented by the prediction of its first token.

4. **Handling Special Tokens:** Special tokens (like padding or segment separators), which do not correspond to any word in the text, are ignored during the alignment process.

5.2.5 Evaluation of Model Performance

Overview of the Evaluation Process

The evaluation function in ILA dynamically assesses the model's effectiveness by comparing predicted labels against true labels across a given set of evaluation documents. This process involves several key steps:

1. **Loading Configuration and Data:** The system retrieves necessary configuration settings such as the model and tokenizer paths. It loads the documents designated for evaluation, either from the project's dataset or an optional supplementary dataset:

- If a supplementary dataset is provided, it performs a full evaluation using this external dataset .
- If no supplementary dataset is available, the system uses the project's existing evaluation documents.

2. **Model and Tokenizer Preparation:** The designated tokenizer is used to process the text, ensuring that it is in the proper format for the model to perform predictions. The model, loaded from its last saved state, is then ready to make predictions.

3. **Prediction and Alignment:** The model predicts labels for each document, and these labels are aligned with the original text to ensure they match the true data structure. This alignment is crucial for accurate performance measurement.

4. **Performance Metrics Calculation:** The F_1-score performance is evaluated to assess the model's performance.

Storing and Reporting Evaluation Results

Once the evaluation is complete, the results are stored in a structured format within the project's database. This allows for tracking the model's performance over time and adjusting training procedures if necessary. The average confidence of predictions, support of labels and other relevant metrics are also recorded to provide a comprehensive view of the model's current state.

5.3 Practical Information

5.3.1 Tool Interface and Usage

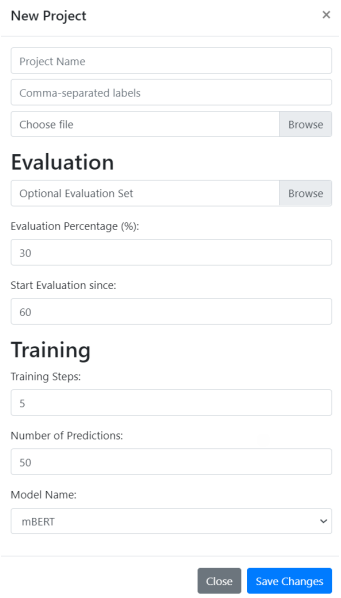


Figure 5.7: Project Creation of ILA.

Section 1: New Project Creation

Allows users to name the project, enter comma-separated labels, and upload the dataset that will be annotated.

Section 2: Evaluation

Enables configuration of an optional evaluation set, percentage of documents used for evaluation, and specifies when evaluation starts.

Section 3: Training

Settings for training include the number of steps, predictions, and selecting the model type (e.g., *mBERT*).

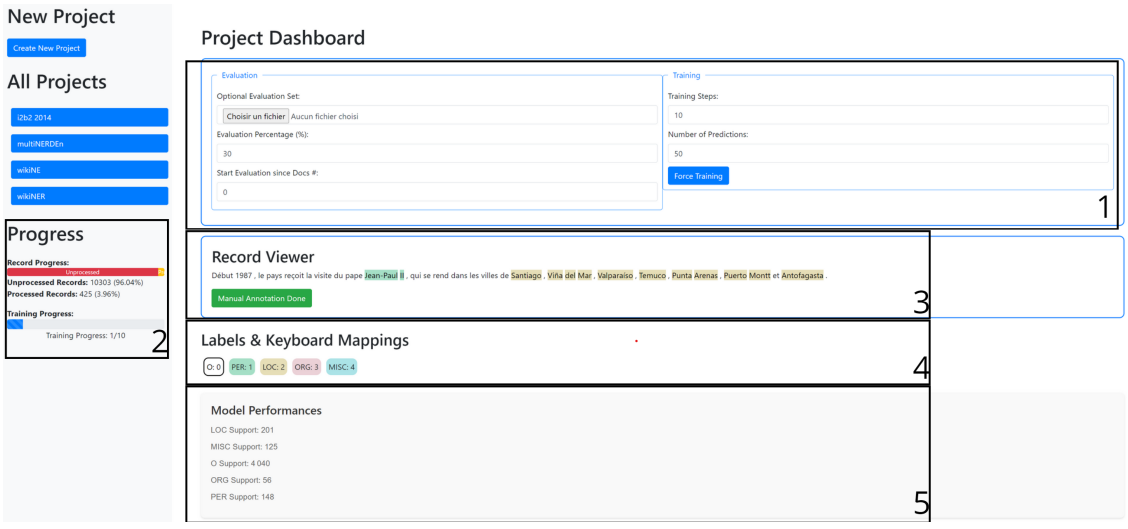


Figure 5.8: Main Page for Record Annotation

Section 1: Parameters Reviewing

During the annotation process, all parameters can be modified, and the model can be manually trained.

Section 2: Project Progress Tracking

This section tracks and displays the progress of the project in terms of records processed, remaining, and the overall training status of the model. It provides essential insights into the workload completed and what still needs to be addressed.

Section 3: Annotation Process

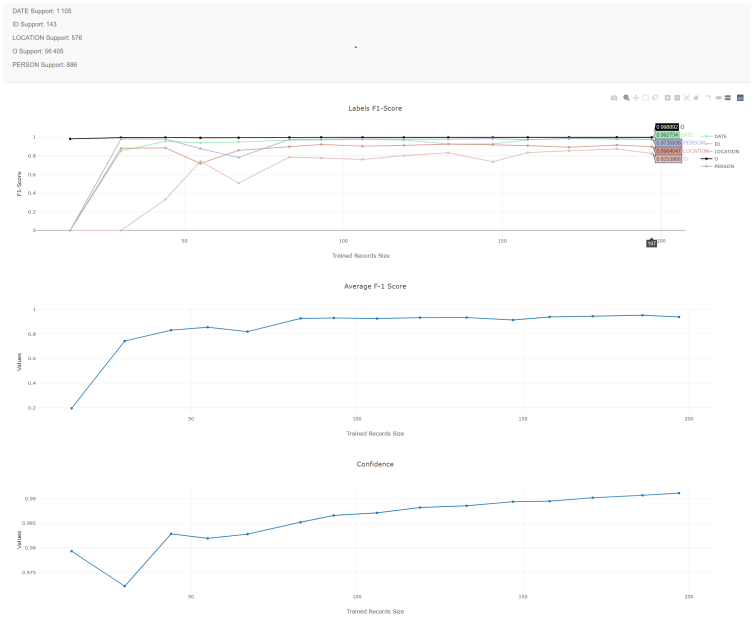
Here, the records appear with the predictions to be annotated.

Section 4: Labels and Keyboard Mappings

This part of the dashboard provides a quick reference for the labels that can be applied to the text data and their corresponding keyboard shortcuts. Enhancing usability, this feature speeds up the manual annotation process by allowing for quick label assignments.

Section 5: Support of labels for Evaluation

This feature enables the user to assess the robustness of the evaluations by indicating the label support within the evaluation set.



The first plot displays F_1-scores for all labels. The second plot depicts average Macro F_1-score, demonstrating model accuracy as data increase. The third plot shows the model's prediction confidence, reflecting reliability across training sizes. Together, these plots provide a compact yet thorough evaluation of the model's capabilities.

Figure 5.9: Evaluation Page of ILA

5.3.2 API Access and Integration

Web-Based Interaction:

The Flask framework underpins the web application, providing a robust, lightweight platform for handling web requests. This setup enables users to manage projects, process data, and view model performance through a user-friendly web interface. The integration of Flask allows for scalable and secure interactions, supporting features such as:

- **Dynamic Data Management:** Users can easily upload new datasets, initiate data processing tasks, and manage annotations directly through the web interface.
- **Real-Time Updates:** The system provides live feedback on the progress of data processing and model training, enhancing the user's ability to monitor and control the workflow.

Command-Line Interface

Direct Script Execution: In addition to the web interface, the tool supports direct interactions through the command line. This feature is particularly useful for users who require automated or batch processing capabilities. Users can initiate processes such as training a model or evaluate data by executing a Python script with command-line arguments, which provide the necessary configurations. The user can use the manager or call each function separately.

```
# With manager
python src/process_new_records.py --project_name wikiNER
# Manual use
python src/train.py --project_name wikiNER
python src/prediction.py --project_name wikiNER --num_records 100
python src/evaluation.py --project_name wikiNER
```

5.4 Results

5.4.1 Evaluation Strategy

In a subsequent experiment, the performance of the evaluation schema was validated using the i2b2 2014 dataset. For this experiment, 30% of the documents were annotated to test the effectiveness of the model under partially labeled conditions.

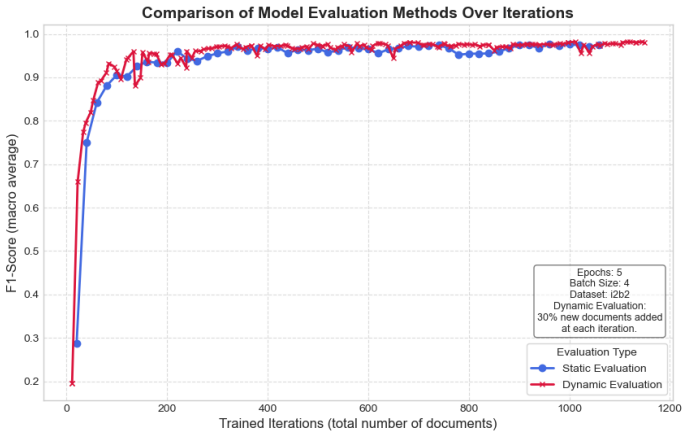


Figure 5.10: Comparison of Evaluation Methods Over Iterations using the i2b2 2014 dataset

The experiment was designed to compare two different evaluation methods:

- **Static Evaluation:** Where the model is evaluated against a fixed set of test documents (test set provided by i2b2 2014.)
- **Dynamic Evaluation:** Where 30% of new documents are added at each iteration, mimicking a real-world scenario where new data become available over time.

The results show that the dynamic evaluation strategy, which continuously incorporates more documents, provides a robust method for performance evaluation, yielding results comparable to static evaluation.

5.4.2 Document Selection Strategies

Then, the focus shifted to the efficacy of training the model by prioritizing documents where the model's predictions had the lowest confidence. The hypothesis was based on the fact that by addressing the most uncertain cases early in the training process, the model could improve more rapidly, especially in its ability to generalize across the dataset.

Experimental Setup

Two document selection strategies were implemented to train the model:

- **Random Document Selection:** Documents are selected randomly from the training set for each training iteration.
- **Lowest Confidence Document Selection:** Documents for which the model's predictions have the lowest confidence scores are prioritized in the training queue.

The average confidence for a model's predictions is calculated using:

$$\text{average_confidence} = \frac{1}{N} \sum_{i=1}^{N} c_i \tag{5.1}$$

where N is the number of predictions, and c_i are the confidence scores derived from the softmax probabilities, reflecting the model's certainty.

Results

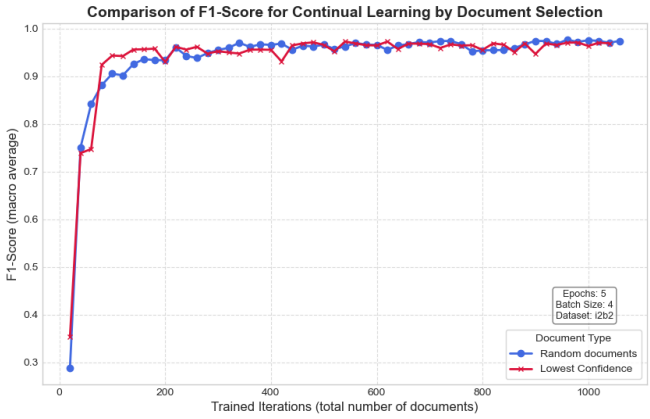


Figure 5.11: Global View of Document Selection Impact

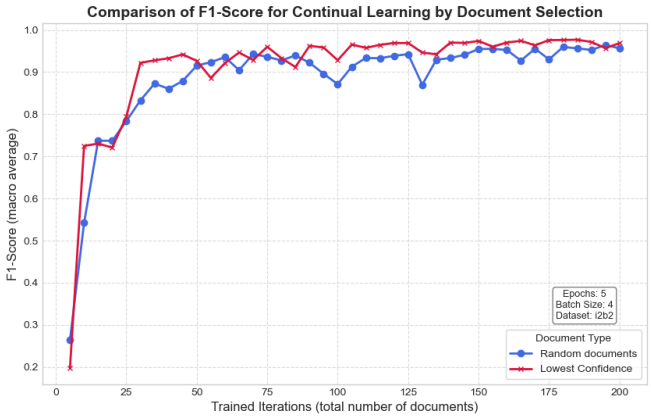


Figure 5.12: Zoomed View over the First Documents

For this process, the model predicts the next 300 documents, retrains itself every 10 documents, and utilizes the *RoBERTa* model. This result clearly shows that the lowest confidence strategy learns more quickly than the random strategy.

5.4.3 Confidence of the Model

To understand if the model's confidence in its predictions increases with more training, the confidence scores were tracked across iterations. Confidence here is measured as the average highest probability assigned to the predicted labels, providing an indicator of the model's certainty.

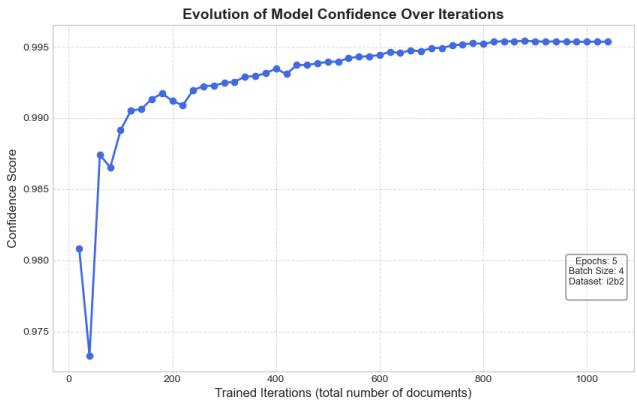


Figure 5.13: Evolution of Model Confidence Over Iterations

This illustrates a clear trend of increasing confidence as the model processes more documents, stabilizing at higher levels as training progresses. When the confidence reaches a plateau, it suggests that the model's learning may have been completed. This indicates successful learning and generalization capabilities.

5.4.4 Impact of Training Data Quality on Model Performance

This evaluation is crucial to determine whether it is necessary to ensure all PHI are identified without missing any.

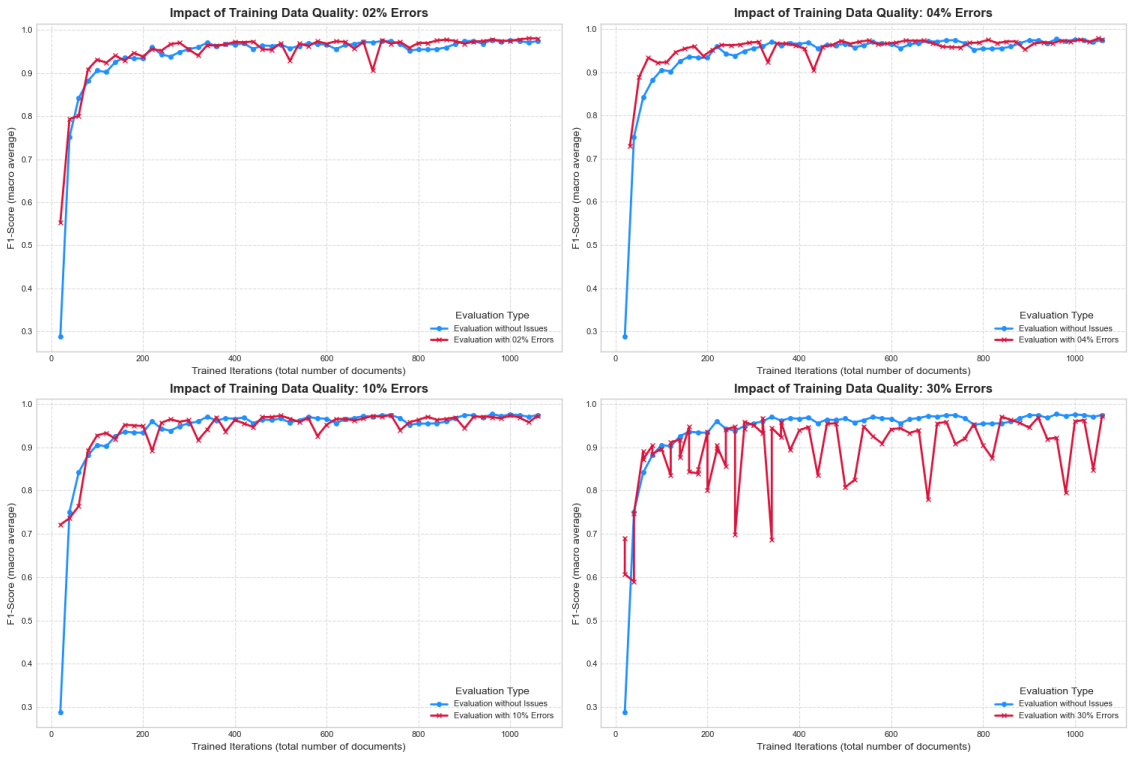


Figure 5.14: Impact of Label Completeness on Model Performance

Experimental Setup and Methodology for Label Omission

To evaluate the impact of label incompleteness on the annotation process, the dataset was manipulated to create several versions where 0%, 2%, 4%, 10%, and 30% of the labels were randomly omitted. This setup allows us to mimic scenarios with varying degrees of label incompleteness.

The methodology for simulating label omission involved the following steps:

- All non-'O' labels in the dataset, representing entities of interest, were first identified.
- From this subset, a specified percentage of labels was randomly selected and replaced with the 'O' label, effectively omitting them from the training data. This method ensures that the simulation reflects a realistic degradation of data quality.

The results show that as the percentage of label omissions increases from 0% to 30%, the model's performance fluctuations become more pronounced. Models trained with higher label omissions exhibit slower convergence and greater instability.

5.4.5 Selective Model Strategy

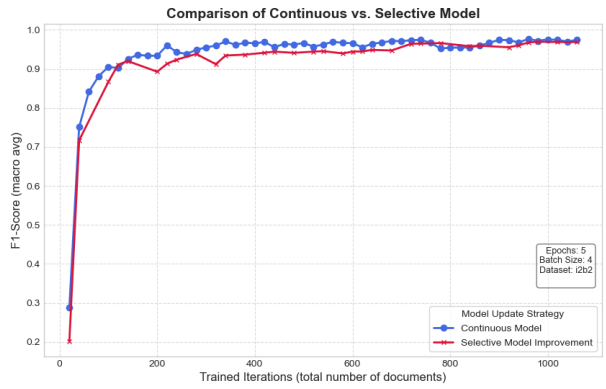


Figure 5.15: Continuous vs. Selective Model Improvement Strategies.

To optimize model performance, a selective improvement strategy was implemented, retaining only updates that enhance the F_1-score. This avoids overfitting on new, potentially noisy data and helps navigate out of local maxima by rejecting detrimental updates. If a new iteration underperforms, it was set as the new baseline, so subsequent models only need to outperform this lower benchmark, increasing the chances of escaping local maxima.

The figure illustrates the comparison between continuous and selective model improvement strategies. The continuous model consistently outperforms the selective model improvement approach across training iterations, achieving higher F_1-scores. This indicates that retaining all updates, rather than selectively keeping only improvements, leads to better overall model performance.

5.5 Discussion

The first step before proceeding further was to evaluate the evaluation strategy to check if it was consistent and if it could be used later for analyses.

5.5.1 Evaluation Strategy and Results

The experimental outcomes, as shown in Figure 5.10, confirm the effectiveness of the dynamic evaluation method. Initially, there is a slight overestimation in model performance. To address this, early results can be ignored while the model's F_1-score increases exponentially and label support remains low. The model's reliability is best indicated when its performance stabilizes.

- **Initial Overestimation:** The dynamic evaluation method initially shows a higher F_1-score compared to the static evaluation. However, this overestimation diminishes as the size of the evaluation set increases.
- **Convergence:** Over time, the F_1-scores from both evaluation methods converge, demonstrating that the model adapts effectively.

Having established a reliable evaluation strategy, the focus then shifted to enhancing training efficiency by addressing model uncertainty. The next step was to investigate if prioritizing low-confidence documents could improve training efficiency and model performance.

5.5.2 Lowest Confidence First

This part tested a novel document selection method based on model uncertainty, prioritizing low-confidence documents to improve training efficiency.

- **Immediate Improvement:** As shown in Figure 5.12, there was a noticeable performance boost, especially for the initial documents (20 to 50).
- **Enhanced Robustness:** This method improved immediate performance and overall robustness, enabling the model to handle diverse data better.
- **High Label Density:** The initial documents had a high label density, which helped the model manage complex labels early.

5.5.3 Model Confidence Over Iterations

The evolution of model confidence, as shown in Figure 5.13, demonstrates an initial exponential growth followed by a plateau. This indicates the model's learning phase, after which confidence in predictions stabilizes. This plateau suggests that the model has effectively learned from the training data and reached its peak performance. Understanding this pattern allows for the determination that the model is reliable once it reaches the plateau phase.

5.5.4 Impact Analysis of Label Omission and Model Robustness

This section evaluates the model's performance under varying levels of label completeness to mimic real-world conditions where perfect data annotation is often unattainable. By systematically reducing label completeness during training, the model's resilience to data quality degradation is assessed. Figure 5.14 depicts the relationship between different levels of label omissions and the corresponding model performance, as measured by the F_1-score.

Cost of Comprehensive Annotation Ensuring well-annotated documents is costly. Carrell et al. [8] analyzed the costs of discovering new PHI instances based on the number of annotators:

- Individual annotators discover a median of 1,056 new PHI instances at \$0.71 per instance.
- Teams of 2 annotators find 31 new PHI instances at \$24.32 per instance.
- Teams of 3 annotators discover 4 new PHI instances at \$188.49 per instance.

- Teams of 4 annotators find 2 new PHI instances at $376.99 per instance.

This analysis highlights the high cost of ensuring complete PHI removal. Given these expenses, it was examined whether institutions need to maintain perfectly annotated datasets for training deep learning models.

Performance Observations The plots demonstrate that label omissions lead to increased instability in F_1-scores. However, the model exhibits a strong ability to quickly recover high performance levels, even with significant label errors. For 2% and 4% error rates, the model's F_1-scores show minor fluctuations but consistently return to high performance after initial instability. At a 10% error rate, the fluctuations become more pronounced, yet the model still maintains high F_1-scores with occasional drops. The most significant impact is observed at a 30% error rate, where the performance shows frequent and deeper drops, indicating that while the model can reach high F_1-scores intermittently, its robustness is significantly compromised. This suggests that label errors affect the stability of the model more than its overall performance, except at very high error rates where both stability and performance are impacted.

These results indicate that using multiple annotators is not necessary to achieve good performance. A single annotator is sufficient, as evidenced by Carrell et al., who found that a single annotator has an error rate of approximately 3.39%. This level of label omission still allows the model to maintain high performance and robustness.

5.5.5 Discussion on Selective Model Improvement Strategy

Implementing a selective model improvement strategy, where updates were only retained if they demonstrated immediate performance benefits, initially appeared to be a prudent approach. This strategy was intended to prevent the integration of potentially detrimental updates from omitted labeled data.

Challenges Encountered

- **Short-term vs. Long-term Performance:** The selective strategy focused on short-term gains, often at the expense of long-term learning and generalization. This resulted in the rejection of updates that did not show immediate improvement but were beneficial for future generalization.
- **Stagnation in Model Learning:** This strategy led to periods of stagnation where no apparent immediate improvements were made, indicating that some iterations, though not immediately improving the F_1-score, are crucial for learning complex patterns.

Inefficacy of the Strategy The test results demonstrate that retaining only the updates that show immediate improvement leads to less optimal overall performance.

CHAPTER 6

CONCLUSION

6.1 Summary of Findings

This study conducted an in-depth exploration of deep learning techniques for the de-identification of clinical documents, leading to several critical insights and the development of innovative solutions.

6.1.1 Challenges with Traditional and Transfer Learning Approaches

Initial efforts using traditional deep learning configurations and transfer learning were met with significant challenges. Traditional models, while performing adequately in controlled experiments, required extensive annotated data, making them impractical for diverse clinical settings where such data is scarce. Transfer learning, intended to leverage pre-trained models, also fell short due to difficulties in adapting to the nuances of specific datasets and generalizing across different data contexts. These limitations highlighted issues such as unstructured and diverse data formats, language diversity, and sparse, inconsistent labeling, all of which hindered the practical application of these models.

6.1.2 Innovative Solution: Incremental Learning Annotator (ILA)

In response to the aforementioned challenges, the Incremental Learning Annotator (ILA) was developed. ILA addresses the need for rapid and efficient model training with limited annotated documents. It enables high-performance model training using significantly fewer examples, facilitating faster deployment and optimization of de-identification models. This innovation is crucial for adapting quickly to the specific needs of clinical document de-identification, ensuring effective implementation even in environments with limited annotated data.

6.1.3 Comparative Analysis and Findings

Through extensive evaluation, this study compared two approaches: Single-Dataset Focus and Cross-Dataset Application. It was found that transfer learning method is inadequate for reliable cross-dataset performance due to issues like unstructured data formats and language diversity. The ILA, however, effectively mitigates these issues by enabling rapid, high-quality model training with fewer annotated examples working in a single dataset.

6.2 Contributions to the Field

This study advances the field of clinical document de-identification by providing open-source tools and methodologies that enhance accessibility, adaptability, and efficiency.

6.2.1 Development of Open-Source Tools

This study provided versatile open-source code supporting both traditional and transfer learning de-identification solutions. This code is adaptable to various data formats, including *csv*, *txt*, *brat*, and *CoNLL-U*, allowing integration into existing workflows. Comprehensive evaluation methods are included, enabling users to assess and refine their de-identification solutions based on robust empirical feedback.

6.2.2 Innovative Incremental Learning Annotator (ILA)

A pivotal contribution is the open-source ILA, which significantly enhances the efficiency of developing and deploying high-performance de-identification models. The ILA leverages incremental learning for continual model improvement, reducing the need for retraining from scratch and enabling rapid adaptation to new data and evolving regulatory requirements. This capability reduces the time and resources required, making advanced de-identification technologies more accessible, even to entities with limited computational resources or specialized expertise. The broader accessibility democratizes the use of cutting-edge NER technologies, facilitating their integration into various healthcare information systems and enhancing the protection of sensitive patient data.

By making this tool open source and easily integrable, this study supports ongoing improvements and encourages wider adoption of effective de-identification practices.

6.3 Limitations and Challenges

This study, while making advances in the field of de-identification using deep learning, has encountered specific limitations that could impact its broader application and effectiveness. These challenges are important for understanding the scope and potential areas for improvement in future work. Below, the primary limitations identified during the research are outlined.

6.3.1 Tokenization

A limitation of this study is that it did not evaluate how variations in preprocessing methods might affect model performance. The study consistently used the default preprocessing method provided with the model, without exploring other options.

6.3.2 Token-Level Assessment

This study emphasizes the identification of individual tokens rather than the grouping and replacement of entire entities. A limitation of this approach is the exclusion of the BIO-Tag (Begin, Inside, Outside Tagging) from the dataset evaluation, which is essential for recognizing and linking tokens into coherent entities.

Focusing on token-level identification necessitates additional steps to transform recognized tokens into complete entities for proper anonymization. Without this transformation, the model's ability to effectively anonymize data is constrained.

6.3.3 Lack of Diversity in ILA Evaluation

A primary limitation in the evaluation of the ILA is the reliance on a limited range of datasets, primarily the i2b2 2014 dataset. This constraint raises concerns about the generalizability of the ILA across different types of clinical documents and datasets. The effectiveness of the ILA in diverse real-world settings remains to be thoroughly tested, potentially limiting confidence in its universal applicability.

6.3.4 User Interface Limitations of the ILA

The WebUI developed for the Incremental Learning Tool, intended to facilitate user interaction and model management, lacks of intuitivity. This shortfall could deter users, especially those with limited technical expertise, from fully utilizing the capabilities of the ILA. Improving the user interface is essential to ensure that the tool is accessible and user-friendly for all potential users, regardless of their technical background.

6.4 Recommendations for Future Research

Building upon the findings and limitations identified in this study, the following recommendations are proposed to guide future research in enhancing the effectiveness and applicability of de-identification tools in clinical document processing. These recommendations address both technical advancements and practical implementations to improve the current state of the art.

6.4.1 Testing with Advanced Deep Learning Models

Future studies should explore the potential of recent advancements in deep learning models, such as Llama 3, Mixtral, Cohere, or Claud, within the frameworks developed in this study. Applying these cutting-edge models could potentially achieve near-perfect performance metrics, such as a 100% F_1-score.

6.4.2 Quantitative Evaluation of Tool Efficiency And Time winning

It is crucial to evaluate the efficiency gains provided by the ILA. Future research should focus on quantifying the time saved per document when using the ILA compared to traditional methods. This evaluation should include detailed metrics on processing times and resource utilization to provide a comprehensive understanding of the tool's practical benefits in real-world scenarios.

6.4.3 Expansion of Evaluation Datasets

To ensure the generalizability of de-identification tools, future work should include a broader array of evaluation datasets from diverse clinical settings and languages. Collaborations with healthcare institutions globally would be beneficial to gather a wide range of clinical documents, thus enhancing the robustness and applicability of the ILA.

6.4.4 Improving Code Modularity and Integration

Improving the modularity and integration capability of the open-source tools provided is essential. Future efforts should focus on refining the architecture to allow for easier integration with existing health information systems, facilitating widespread adoption and customization as per user needs.

By addressing these recommendations, future research can significantly advance the field of clinical document de-identification, improving both the technology and its implementation in practice, thereby ensuring that patient privacy is maintained while maximizing the utility of medical data for research and treatment.

6.5 ChatGPT

For this study, the services of ChatGPT were used to obtain well-formulated sentences. All the content of this work has first been completely written with basic english words and "franglais" expressions. Then, ChatGPT helped identify and reformulate sentences or paragraphs that lacked of clarity.

Appendix A

A.1 Mappings

A.1.1 Meddocan Mapping

```
meddocan_mapping = {
    "B-SEXO_SUJETO_ASISTENCIA": {"mapped": "O", "support": 1334 + 446},
    "B-FAMILIARES_SUJETO_ASISTENCIA": {"mapped": "O", "support": 323 + 77}
,
    "I-FAMILIARES_SUJETO_ASISTENCIA": {"mapped": "O", "support": 171 + 53}
,
    "B-ID_CONTACTO_ASISTENCIAL": {"mapped": "O", "support": 98 + 36},
    "I-HOSPITAL": {"mapped": "LOCATION", "support": 1236 + 402},
    "B-EDAD_SUJETO_ASISTENCIA": {"mapped": "O", "support": 1552 + 518},
    "B-CALLE": {"mapped": "LOCATION", "support": 1292 + 412},
    "I-PAIS": {"mapped": "LOCATION", "support": 12 + 3},
    "B-NOMBRE_PERSONAL_SANITARIO": {"mapped": "PERSON", "support": 1497 +
501},
    "I-EDAD_SUJETO_ASISTENCIA": {"mapped": "O", "support": 1513 + 497},
    "I-CORREO_ELECTRONICO": {"mapped": "O", "support": 14 + 14},
    "B-FECHAS": {"mapped": "DATE", "support": 1905 + 594},
    "B-PAIS": {"mapped": "LOCATION", "support": 974 + 327},
    "I-TERRITORIO": {"mapped": "LOCATION", "support": 1016 + 327},
    "I-NOMBRE_PERSONAL_SANITARIO": {"mapped": "PERSON", "support": 3223 +
1092},
    "B-INSTITUCION": {"mapped": "LOCATION", "support": 146 + 52},
    "I-NUMERO_FAX": {"mapped": "ID", "support": 24 + 8},
    "B-CORREO_ELECTRONICO": {"mapped": "O", "support": 689 + 238},
    "I-CENTRO_SALUD": {"mapped": "LOCATION", "support": 32 + 26},
    "I-NOMBRE_SUJETO_ASISTENCIA": {"mapped": "PERSON", "support": 821 + 27
1},
    "I-SEXO_SUJETO_ASISTENCIA": {"mapped": "O", "support": 3 + 0},
    "B-PROFESION": {"mapped": "O", "support": 26 + 9},
```

```
    "B-ID_SUJETO_ASISTENCIA": {"mapped": "ID", "support": 801 + 257},
    "B-TERRITORIO": {"mapped": "LOCATION", "support": 2113 + 701},
    "B-NOMBRE_SUJETO_ASISTENCIA": {"mapped": "PERSON", "support": 1511 + 5
02},
    "I-CALLE": {"mapped": "LOCATION", "support": 5071 + 1631},
    "B-OTROS_SUJETO_ASISTENCIA": {"mapped": "O", "support": 14 + 7},
    "I-INSTITUCION": {"mapped": "LOCATION", "support": 368 + 132},
    "I-ID_TITULACION_PERSONAL_SANITARIO": {"mapped": "ID", "support": 1313
 + 430},
    "B-ID_EMPLEO_PERSONAL_SANITARIO": {"mapped": "ID", "support": 1 + 0},
    "B-CENTRO_SALUD": {"mapped": "LOCATION", "support": 8 + 6},
    "I-ID_ASEGURAMIENTO": {"mapped": "O", "support": 1098 + 377},
    "B-ID_ASEGURAMIENTO": {"mapped": "O", "support": 585 + 198},
    "I-ID_SUJETO_ASISTENCIA": {"mapped": "ID", "support": 43 + 7},
    "I-ID_EMPLEO_PERSONAL_SANITARIO": {"mapped": "ID", "support": 3 + 0},
    "I-PROFESION": {"mapped": "O", "support": 34 + 12},
    "B-NUMERO_FAX": {"mapped": "ID", "support": 20 + 6},
    "I-NUMERO_TELEFONO": {"mapped": "ID", "support": 121 + 41},
    "B-NUMERO_TELEFONO": {"mapped": "ID", "support": 81 + 25},
    "I-FECHAS": {"mapped": "DATE", "support": 525 + 156},
    "B-ID_TITULACION_PERSONAL_SANITARIO": {"mapped": "ID", "support": 697
+ 234},
    "B-HOSPITAL": {"mapped": "LOCATION", "support": 390 + 128},
    "I-OTROS_SUJETO_ASISTENCIA": {"mapped": "O", "support": 32 + 5},
    "O": {"mapped": "O", "support": 286335 + 94304}
}
```

A.1.2 i2b2 2006 Mapping

```
i2b2_2006Mapping = {
    "I-HOSPITAL": {"mapped": "LOCATION", "support": 1879 + 926},
    "B-HOSPITAL": {"mapped": "LOCATION", "support": 1724 + 676},
    "I-LOCATION": {"mapped": "LOCATION", "support": 216 + 121},
    "B-LOCATION": {"mapped": "LOCATION", "support": 144 + 119},
    "I-DATE": {"mapped": "DATE", "support": 323 + 230},
    "B-DATE": {"mapped": "DATE", "support": 5136 + 1925},
    "I-DOCTOR": {"mapped": "PERSON", "support": 3698 + 1242},
    "B-DOCTOR": {"mapped": "PERSON", "support": 2681 + 1070},
    "I-PATIENT": {"mapped": "PERSON", "support": 1117 + 268},
    "B-PATIENT": {"mapped": "PERSON", "support": 684 + 245},
    "I-PHONE": {"mapped": "ID", "support": 62 + 27},
    "B-PHONE": {"mapped": "ID", "support": 174 + 58},
    "I-ID": {"mapped": "ID", "support": 253 + 57},
    "B-ID": {"mapped": "ID", "support": 3635 + 1137},
    "B-AGE": {"mapped": "O", "support": 13 + 3},
    "I-AGE": {"mapped": "O", "support": 0},
```

```
    "O": {"mapped": "O", "support": 368648 + 156506}
}
```

A.1.3 i2b2 2014 Mapping

```
i2b2_2014Mapping = {
    "I-COUNTRY": {"mapped": "LOCATION", "support": 8 + 13},
    "B-COUNTRY": {"mapped": "LOCATION", "support": 65 + 117},
    "I-CITY": {"mapped": "LOCATION", "support": 89 + 82},
    "B-CITY": {"mapped": "LOCATION", "support": 391 + 252},
    "I-STREET": {"mapped": "LOCATION", "support": 433 + 280},
    "B-STREET": {"mapped": "LOCATION", "support": 216 + 136},
    "I-ZIP": {"mapped": "LOCATION", "support": 0 + 0},
    "B-ZIP": {"mapped": "LOCATION", "support": 211 + 140},
    "I-STATE": {"mapped": "LOCATION", "support": 3 + 15},
    "B-STATE": {"mapped": "LOCATION", "support": 310 + 189},
    "I-HOSPITAL": {"mapped": "LOCATION", "support": 1371 + 708},
    "B-HOSPITAL": {"mapped": "LOCATION", "support": 1348 + 823},
    "I-LOCATION-OTHER": {"mapped": "LOCATION", "support": 8 + 7},
    "B-LOCATION-OTHER": {"mapped": "LOCATION", "support": 4 + 13},
    "I-ORGANIZATION": {"mapped": "LOCATION", "support": 112 + 60},
    "B-ORGANIZATION": {"mapped": "LOCATION", "support": 124 + 80},
    "I-DATE": {"mapped": "DATE", "support": 1053 + 614},
    "B-DATE": {"mapped": "DATE", "support": 6579 + 4419},
    "I-DOCTOR": {"mapped": "PERSON", "support": 2102 + 1326},
    "B-DOCTOR": {"mapped": "PERSON", "support": 3599 + 1793},
    "I-PATIENT": {"mapped": "PERSON", "support": 693 + 433},
    "B-PATIENT": {"mapped": "PERSON", "support": 1310 + 875},
    "I-MEDICALRECORD": {"mapped": "ID", "support": 30 + 17},
    "B-MEDICALRECORD": {"mapped": "ID", "support": 487 + 348},
    "I-PHONE": {"mapped": "ID", "support": 51 + 49},
    "B-PHONE": {"mapped": "ID", "support": 210 + 155},
    "I-FAX": {"mapped": "ID", "support": 2 + 1},
    "B-FAX": {"mapped": "ID", "support": 8 + 2},
    "I-IDNUM": {"mapped": "ID", "support": 11 + 8},
    "B-IDNUM": {"mapped": "ID", "support": 152 + 97},
    "I-USERNAME": {"mapped": "ID", "support": 0 + 0},
    "B-USERNAME": {"mapped": "ID", "support": 95 + 92},
    "I-HEALTHPLAN": {"mapped": "O", "support": 1 + 0},
    "B-HEALTHPLAN": {"mapped": "O", "support": 1 + 0},
    "I-DEVICE": {"mapped": "O", "support": 0 + 2},
    "B-DEVICE": {"mapped": "O", "support": 7 + 8},
    "B-AGE": {"mapped": "O", "support": 1197 + 758},
    "I-AGE": {"mapped": "O", "support": 14 + 3},
    "I-PROFESSION": {"mapped": "O", "support": 188 + 159},
    "B-PROFESSION": {"mapped": "O", "support": 230 + 168},
```

```
        "I-EMAIL": {"mapped": "O", "support": 0 + 0},
        "B-EMAIL": {"mapped": "O", "support": 4 + 1},
        "I-URL": {"mapped": "O", "support": 4 + 0},
        "B-URL": {"mapped": "O", "support": 2 + 0},
        "B-BIOID": {"mapped": "O", "support": 1 + 0},
        "I-BIOID": {"mapped": "O", "support": 0 + 0},
        "O": {"mapped": "O", "support": 467298 + 301969}
    }
```

A.1.4 multiNERD Mapping

```
    multiNERDMapping = {
        "I-PER": {"mapped": "PERSON", "support": 66188 + 5730},
        "B-PER": {"mapped": "PERSON", "support": 62987 + 5265},
        "I-LOC": {"mapped": "LOCATION", "support": 24400 + 5963},
        "B-LOC": {"mapped": "LOCATION", "support": 58665 + 12024},
        "B-TIME": {"mapped": "DATE", "support": 2540 + 289},
        "I-TIME": {"mapped": "DATE", "support": 1971 + 208},
        "ORG": {"mapped": "O", "support": 27641 + 3309},
        "ANIM": {"mapped": "O", "support": 12736 + 1604},
        "MEDIA": {"mapped": "O", "support": 6081 + 458},
        "DIS": {"mapped": "O", "support": 8702 + 759},
        "FOOD": {"mapped": "O", "support": 8279 + 566},
        "PLANT": {"mapped": "O", "support": 7436 + 894},
        "EVE": {"mapped": "O", "support": 2525 + 352},
        "CEL": {"mapped": "O", "support": 2685 + 41},
        "VEHI": {"mapped": "O", "support": 404 + 32},
        "INST": {"mapped": "O", "support": 379 + 12},
        "MYTH": {"mapped": "O", "support": 569 + 32},
        "BIO": {"mapped": "O", "support": 140 + 8},
        "O": {"mapped": "O", "support": 2478599 + 301442}
    }
```

A.2 Roberta Code

```
class RobertaCustomForTokenClassification(nn.Module):
    def __init__(self, num_labels, config=None):
        super(RobertaCustomForTokenClassification, self).__init__()

        if isinstance(num_labels, RobertaConfig):
            self.config = num_labels
        else:
```

```
        self.config =
RobertaConfig.from_pretrained("Jean-Baptiste/roberta-large-ner-english",
num_labels=num_labels)

    self.roberta =
RobertaModel.from_pretrained("Jean-Baptiste/roberta-large-ner-english",
config=self.config)
    self.classifier = nn.Linear(self.config.hidden_size,
self.config.num_labels)

def forward(self, input_ids=None, attention_mask=None, labels=None,
threshold=0.5):
    outputs = self.roberta(input_ids=input_ids,
attention_mask=attention_mask)
    logits = self.classifier(outputs[0])

    probabilities = F.softmax(logits, dim=-1)
    label_predictions = (probabilities > threshold).long()
    loss = None
    if labels is not None:
        loss_fct = nn.CrossEntropyLoss()
        active_loss = attention_mask.view(-1) == 1 if attention_mask is
not None else torch.tensor([True]*logits.numel())
        active_logits = logits.view(-1,
self.config.num_labels)[active_loss]
        active_labels = labels.view(-1)[active_loss]
        loss = loss_fct(active_logits, active_labels)
    return {'loss': loss, 'logits': logits, 'label_predictions':
label_predictions}

@classmethod
def from_pretrained(cls, pretrained_dir, *model_args, **kwargs):
    config = RobertaConfig.from_pretrained(pretrained_dir)
    model = cls(config.num_labels, config=config)
    model_path = os.path.join(pretrained_dir, 'pytorch_model.bin')
    state_dict = torch.load(model_path, map_location=torch.device('cuda'))
    model.load_state_dict(state_dict)
    return model

def save_pretrained(self, save_directory):
    if not os.path.exists(save_directory):
        os.makedirs(save_directory)
    config_path = os.path.join(save_directory, 'config.json')
    self.config.to_json_file(config_path)
    torch.save(self.state_dict(), os.path.join(save_directory,
'pytorch_model.bin'))
```

A.3 Description of Machine Learning key techniques

- **Support Vector Machines (SVM)**: SVMs have been particularly effective in classifying text into PHI and non-PHI categories with high accuracy, as demonstrated by studies such as those by Tafvizi et al. [43], which reported a precision-recall balance noteworthy for clinical texts.
- **Conditional Random Fields (CRF)**: CRFs excel in modeling the dependencies and structural patterns in text data, making them ideal for the sequential nature of clinical narratives. Their application has been critical in achieving high F1-scores in PHI identification tasks.
- **Decision Trees**: These models provide transparent decision-making processes, which are invaluable in settings where interpretability is as crucial as accuracy. Their structured approach to decision paths helps in pinpointing specific features that indicate PHI.

A.4 Historical Approaches of Deep Learning

The evolution of neural networks in the field of text de-identification has seen significant advancements with the development of Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Bidirectional LSTMs (Bi-LSTMs).

- **Recurrent Neural Networks (RNNs):** RNNs are effective for sequential data processing, crucial in de-identification of clinical documents. By using previous outputs as inputs, RNNs maintain context, aiding in the recognition and classification of PHI. However, RNNs face significant challenges, notably the vanishing gradient problem, where gradients during training diminish, preventing the model from learning long-distance dependencies. This issue is exacerbated in large documents, making RNNs less ideal for handling extensive clinical texts where distant textual elements are contextually linked.
- **Short-Term Memory Networks (LSTMs):** Long Short-Term Memory Networks (LSTMs) are an extension of RNNs designed to better capture long-term dependencies and avoid issues like the vanishing gradient problem, which hampers learning in standard RNNs. LSTMs have shown significant effectiveness in complex sequence modeling tasks and have been particularly impactful in improving the accuracy of de-identification systems. An example includes the study by Jiang et al. in 2017, where a combination of CRF and LSTM achieved an F1-score of 85.7% [23].
- **Bidirectional LSTMs (Bi-LSTMs):** Bidirectional LSTMs further enhance the capabilities of LSTMs by processing data in both forward and backward directions, providing a richer understanding of context. This architecture is especially powerful in text processing, where the context from both past and future tokens is crucial for accurate interpretation.

- **Transformers:** The introduction of the Transformer model marked a paradigm shift in how text data is processed. Originating from the groundbreaking paper "Attention Is All You Need" by Vaswani et al. in 2017 [48], the model eschews traditional sequential processing typical of RNNs. Instead, it employs a network of self-attention mechanisms that process various parts of input data in parallel, greatly enhancing the training efficiency and effectiveness of the model. One of the most successful implementations of this architecture is BERT (Bidirectional Encoder Representations from Transformers), developed by researchers at Google. BERT revolutionizes task-specific training by pre-training on a vast corpus and then fine-tuning for particular tasks. This technique has led to remarkable improvements in numerous NLP tasks, including the de-identification of PHI. For instance, Johnson's 2020 implementation of the BERT-large model achieved an impressive F1-score of 98.82% [24]. Subsequent advancements, such as ROBERTA, which builds on the foundations laid by BERT, continue to advance the field. This is demonstrated by Meaney et al. (2022), who achieved an F1-score of 96.75% [33], further evidencing the transformative impact of Transformer-based models.

A.5 General Deep Learning Concepts

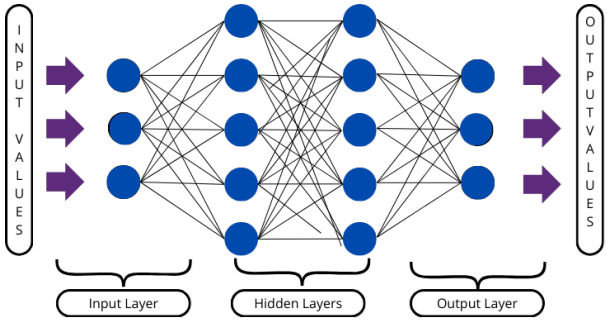


Figure A.1: Deep Learning Representation

This illustrates a basic neural network architecture, which is foundational in deep learning. A neural network consists of an input layer, one or more hidden layers, and an output layer. The input layer receives the raw data that the network will learn from. This data is then fed forward through the hidden layers where the actual processing is done through a series of weighted connections. The hidden layers can be thought of as a series of transformations that convert the input into a format that the output layer can use to make a final prediction or classification.

Each circle in Figure A.1 represents a neuron, and each line represents a connection between neurons across different layers. The strength of these connections is determined by weights, which are adjusted during the training process. The network's ability to learn complex patterns lies in the optimization of these weights.

A.5.1 Understanding Neurons: Weights, Biases, and Activation Functions

Delving deeper into the individual neuron, as depicted in Figure A.2, each neuron receives inputs, which are multiplied by weights—a measure of the strength or importance of the input. A bias is then added to these weighted inputs.

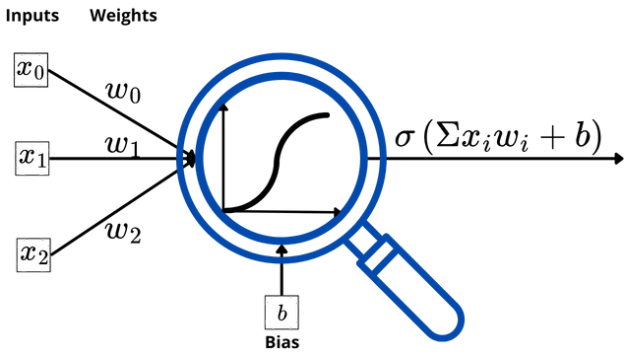


Figure A.2: The internal workings of a neuron, showing inputs being weighted, the addition of a bias, and the activation function being applied.

The weighted sum of the inputs plus the bias forms the net input to the activation function. This function, as the name suggests, determines whether and to what extent a signal should progress further through the network. Three commonly used activation functions are:

- Sigmoid: $f(x) = \frac{1}{1+e^{-x}}$ which maps any real-valued number into the (0, 1) range, making it useful for binary classification.
- Tanh: $f(x) = \tanh(x)$ which also takes any real-valued input and outputs values in the range (-1, 1). It is a rescaled version of the sigmoid, and its output being zero-centered makes it preferred in some cases.
- ReLU: $f(x) = \max(0, x)$, which outputs the input directly if it is positive, otherwise, it outputs zero. It has become the default activation function for many types of neural networks because it helps to reduce the likelihood of vanishing gradient during training.

Activation functions add non-linear properties to the network, enabling it to learn more complex decision boundaries. Without non-linearities, a neural network, regardless of how many layers it had, would behave just like a single-layer perceptron, which can only learn linearly separable functions.

A.5.2 Introduction to Natural Language Processing

In the context of Natural Language Processing (NLP), certain architectures are more prevalent due to the sequential nature of language data. Recurrent architectures like RNNs and LSTMs are

uniquely suited for processing sequences such as sentences or time-series data. Unlike feedforward neural networks, these architectures possess a form of memory that enables them to handle inputs of varying lengths and remember past information, which is critical in understanding the context within language.

A.5.3 Recurrent Neural Network (RNN)

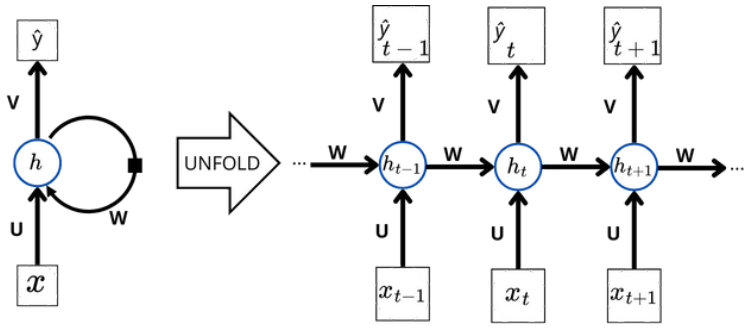


Figure A.3: The basic structure of a Recurrent Neural Network (RNN).

Recurrent Neural Networks (RNNs) contain loops that allow the information to be carried across neurons while processing sequences. This means that the output from a neuron at one time step becomes part of the input to the same neuron at the next time step. This recurrent structure allows the network to maintain a form of state or memory.

However, standard RNNs are prone to two significant issues: the vanishing and exploding gradient problems. These issues make it challenging for the RNN to learn long-distance correlations in a sequence, such as dependencies between words that are far apart in a sentence.

A.5.4 Challenges: Vanishing and Exploding Gradients in RNNs

Recurrent Neural Networks (RNNs) are powerful for modeling sequence data due to their ability to maintain a form of state or memory across inputs. However, RNNs are inherently sensitive to the length of the input sequences they process. This sensitivity arises primarily due to the vanishing and exploding gradient problems during backpropagation through time (BPTT).

Understanding the Gradient Problems In the backpropagation process of RNNs, gradients are computed to update the weights using the chain rule. For a given sequence, the gradient of the loss function with respect to the weights involves terms from every timestep of the sequence. Mathematically, if considering a simple RNN with a weight matrix W and activation function h, the gradient at timestep t is computed as:

$$\frac{\partial L}{\partial W} = \sum_{k=1}^{t} \frac{\partial L}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W}$$

where $\frac{\partial h_t}{\partial h_k}$ is the product of derivatives across timesteps from k to t, each involving W. This term raises two main issues:

- **Vanishing Gradients:** If the eigenvalues of W are less than one, $\frac{\partial h_t}{\partial h_k}$ exponentially shrinks as the sequence length increases, making the gradient increasingly smaller. This causes the network to lose its ability to learn long-range dependencies, as the contributions to the gradient from early inputs practically vanish.

- **Exploding Gradients:** Conversely, if the eigenvalues of W are greater than one, $\frac{\partial h_t}{\partial h_k}$ exponentially grows, leading to very large gradients. This can cause learning to diverge and makes the training unstable.

A.5.5 Mitigation with LSTM Units

Long Short-Term Memory (LSTM) units are designed to overcome these issues by introducing a more complex internal structure with gates that regulate the flow of information. These gates ensure that the gradient can flow through the network without vanishing or exploding, even for long sequences.

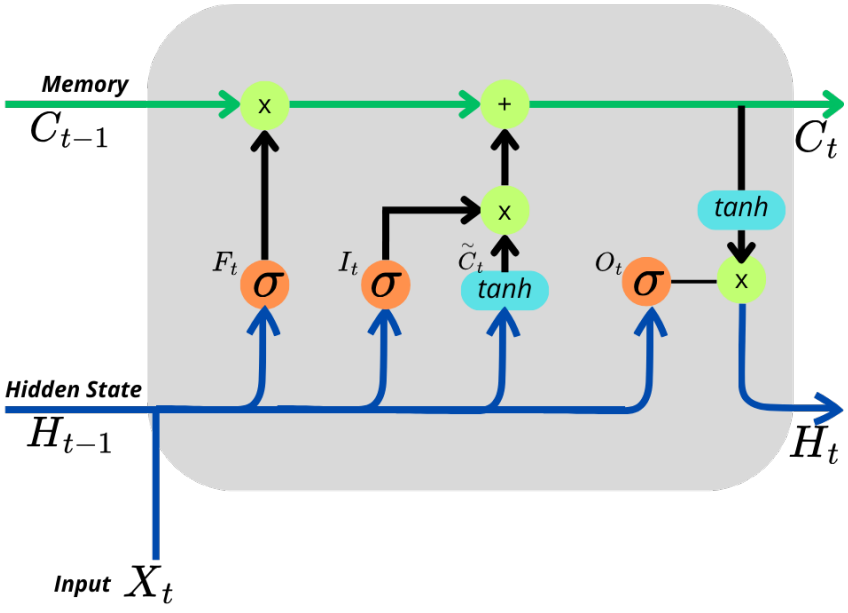


Figure A.4: Diagram illustrating the internal structure of an LSTM unit, including its gates and cell state.

The key components of LSTM that help mitigate these gradient problems include:

- **Forget Gate:** $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ decides what information is discarded from the cell state, allowing the model to "forget" unnecessary information, which helps in controlling the gradient flow.
- **Input Gate:** $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ and the state update $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$ work together to add new information to the cell state, enabling the preservation of important new information without affecting the old information.
- **Output Gate:** $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ controls the output of the cell state to the next layer, or the next timestep, modulating the effect of the cell's memory on the network's output.
- **Cell State Update:** The final cell state is updated as $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$. This combination of forgetting and adding selectively allows the LSTM to maintain stable gradients over time by balancing the rate at which information flows through the network.

Each gate in an LSTM uses the sigmoid activation function σ, limiting the output to a range between 0 and 1, thus providing a way to proportionally allow information through. The architecture of LSTMs thus helps in preserving essential information over long periods, and effectively mitigates the vanishing gradient problem, which is critical for learning long-distance temporal dependencies.

A.5.6 From Sequential to Contextual Processing: RNNs vs. Self-Attention

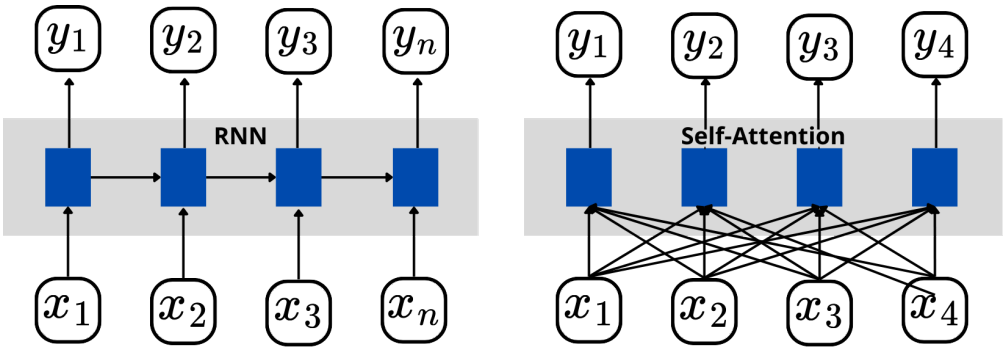


Figure A.5: Comparison between Recurrent Neural Networks (RNNs) and the self-attention mechanism. The left side shows how an RNN processes input sequence step by step, with each output y_i depending on the previous hidden state and the current input x_i. The right side illustrates the self-attention mechanism where each output y_i is computed by attending to all positions within the input sequence, allowing for a more comprehensive understanding of context.

In traditional Recurrent Neural Networks (RNNs), depicted on the left side of Figure A.5, the model processes sequences in a step-by-step manner. Here, the output at any given step y_i is contingent upon the accumulated hidden states up to that point, which are in turn influenced by previous inputs $x_1, x_2, ..., x_{i-1}$. While this allows for temporal dynamic behavior in sequence data, it inherently

restricts the network's capability to learn dependencies for information separated by long distances in the input sequence. This limitation is due to the decay of gradients in backpropagation over time, known as the vanishing gradient problem, and can result in significant challenges when learning the context and structure in language tasks.

To address the limitations inherent to RNNs, the self-attention mechanism, which is the core of the Transformer architecture shown on the right side of Figure A.5, introduces a fundamentally different approach. Self-attention allows each output y_i to be computed by attending to all positions within the input sequence directly, thus, not limited by sequential processing. This mechanism computes attention scores reflecting the influence of all other tokens in the sequence on each token, providing a rich contextual representation.

The Transformer architecture leverages this self-attention mechanism to efficiently process sequences in parallel and capture long-range dependencies, irrespective of sequence length. This parallelization not only speeds up the training process but also solves the vanishing gradient problem by facilitating more direct paths for the gradients to flow through.

Moreover, the Transformer incorporates positional encoding to maintain the order of the sequence, which is a necessity due to the non-sequential nature of self-attention. Positional encodings are added to the input embeddings to provide the model with information about the relative or absolute position of the tokens in the sequence:

$$PE_{(pos,2i)} = \sin\left(pos/10000^{2i/d_{\text{model}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(pos/10000^{2i/d_{\text{model}}}\right)$$

where pos is the position, i is the dimension, and d_{model} is the dimensionality of the model's output.

Through self-attention and positional encoding, the Transformer is able to generate representations of text that capture both the semantics of individual words and the syntactical structure of the entire sequence. As a result, it sets the foundation for the development of complex models like BERT, which can understand and generate human-like text by considering the full context of each word.

A.6 Evaluation

A.6.1 Dataset Format

1. **XML Format:** The XML format, exemplified by the Meddocan dataset (see Subsection 2.1.1), encapsulates the document text at the beginning, followed by annotations containing labels with precise positional data at the end of the file. Each document is stored in a separate file. An example of this format is presented below:

```
<MEDDOCAN>
  <TEXT>
  Nombre: Espagito.
  Apellidos: Rico Magniola.
  ...
  </TEXT>
  <TAGS>
    <NAME id="T21" start="29" end="36" text="Espagito" TYPE="NOMBRE" comment=""/>
    <NAME id="T20" start="49" end="61" text="Rico Magniola" TYPE="NOMBRE" comment=
    ...
  </TAGS>
</MEDDOCAN>
```

XML provides a robust structure for data representation but involves complexity in parsing and processing.

2. **BRAT Format:** Originating from the BRAT annotation tool, this format is widely used for named-entity recognition tasks. It separates the document text and annotations into different files, enhancing clarity but requiring synchronization between text and annotations. Here is an illustrative example:

```
Glutamine synthétase et glutamine gamma-glutamyl transferase
dans le rein de l'homme, du chien et du rat
T1 CHEM 0 20 Glutamine synthétase
#1 AnnotatorNotes T1 C0017801
T2 CHEM 0 9 Glutamine
...
T10 LIVB 103 106 rat
#10 AnnotatorNotes T10 C0034721
```

While BRAT is beneficial for detailed annotations, its split-file structure can complicate data handling and integration in automated systems.

3. **ConLL-U Format:** The ConLL-U format prefixes each line with the word's index in the document, followed by the word and its label:

```
0 word label
1 word label
...
```

This indexing is useful for context recognition in sequential data processing but was deemed unnecessary for our study's scope focused on straightforward label assignment.

4. **TSV Format:** This format simplifies data annotation by aligning each word with its corresponding label in a tab-separated values format.

```
word label
word label
...
```

A.6.2 Grouping Labels Results

Table A.1: Comparison of Model Performance on Meddocan Dataset (Individual Labels)

Label	Support	BIO-Tag F_1	~~BIO-Tag~~ F_1
DOCTOR	834	99.8	99.8
PATIENT	773	99.9	**100.0**
DATE	647	99.2	**99.5**
HOSPITAL	64	94.8	**96.2**
STREET	1283	**99.4**	99.3
COUNTRY	265	98.7	**99.3**
STATE	562	97.9	**98.9**
HEALTH_CENTER	3	0.0	0.0
ORGANIZATION	51	**70.5**	63.7
ID_CONTACT	36	100.0	100.0
MEDRECORD	264	**97.4**	97.0
PHONE	6	100.0	100.0
USERNAME	664	99.8	99.8
GENDER_PATIENT	442	**99.4**	99.0
FAMILY_PATIENT	82	74.7	**80.0**
AGE_PATIENT	1005	**99.4**	99.1
EMAIL	19	64.5	**77.4**
PROFESSION	20	**97.6**	86.4
LOCATION_OTHER	7	0.0	0.0
HEALTHPLAN	575	99.7	**100.0**

Table A.2: Comparison of Model Performance on Meddocan Dataset (Grouped Labels)

Label	Support	BIO-Tag F_1	~~BIO-Tag~~ F_1
PERSON	1607	99.9	**99.9**
DATE	647	99.3	**99.6**
LOCATION	2228	98.7	**99.3**
ID	934	**99.2**	99.1
O	45539	99.9	99.9

Table A.3: Comparison of Model Performance on Meddocan Dataset

Label	Support	Original Labels F_1	Grouped Labels F_1
PERSON	1607	99.9	**99.9**
DATE	647	99.5	**99.6**
LOCATION	2228	98.3	**99.3**
ID	934	99.0	**99.1**
O	45539	98.1	**99.9**

A.6.3 Model Performance on i2b2 2006 Dataset

Table A.4: Performance of different models on i2b2 2006

Label	*RoBERTa*	*XLM-RoBERTa*	*mBERT*	*BERT*	*DistilBERT*	*AlBERT*
DOCTOR	92.48	**93.37**	92.55	91.81	90.98	91.34
PATIENT	93.50	94.20	**94.65**	94.23	93.35	93.44
DATE	**99.51**	99.16	99.25	99.48	98.87	99.06
HOSPITAL	94.08	95.64	**98.16**	94.65	95.35	95.31
PHONE	95.38	**98.51**	91.18	91.67	94.29	94.12
O	99.91	99.36	**99.94**	99.93	99.90	99.90

A.6.4 Model Performance on *MEDDOCAN* Dataset

Table A.5: Performance of different models on Meddocan

Label	*RoBERTa*	*XLM-RoBERTa*	*mBERT*	*BERT*	*DistilBERT*	*AlBERT*
DOCTOR	99.82	100.00	100.00	99.77	99.49	99.82
PATIENT	99.94	99.94	99.94	**100.00**	99.87	99.55
DATE	99.46	99.48	**99.56**	99.39	98.94	99.23
HOSPITAL	96.24	**97.86**	95.52	95.83	92.52	84.29
STREET	**99.30**	99.10	98.80	99.20	98.51	98.96
COUNTRY	**99.25**	98.94	98.06	97.95	98.11	97.91
STATE	**98.93**	98.41	98.81	98.25	97.62	98.06
ORGANIZATION	63.74	**80.22**	78.26	70.69	66.09	57.69
ID_CONTACT	100.0	100.00	98.59	100.00	98.59	98.59
MEDRECORD	96.99	98.31	**98.50**	96.99	97.9	98.09
PHONE	100.0	100.00	100.00	54.55	0.00	0.00
USERNAME	99.77	99.77	99.77	99.55	99.33	99.33
GENDER_PATIENT	98.99	99.55	99.33	**99.77**	99.10	99.32
FAMILY_PATIENT	80.00	79.79	82.41	**83.87**	75.00	69.80
AGE_PATIENT	99.11	98.92	**99.21**	98.46	98.27	98.27
EMAIL	77.42	93.94	**94.20**	72.22	72.22	77.78
PROFESSION	86.36	**86.96**	82.93	66.67	45.16	62.86
HEALTHPLAN	99.91	99.91	99.91	99.91	99.91	99.91
O	99.89	99.91	99.91	99.86	99.82	99.83

A.6.5 Batch Size Performances with i2b2 2006

Table A.6: F1 Scores for Different Batch Size Across Various Configurations i2b2 2006

Label	**1**	**2**	**4**	**8**	**16**	**32**
DOCTOR	96.14	**96.47**	96.04	93.76	94.56	92.51
PATIENT	95.72	**96.58**	96.09	94.80	94.25	94.27
DATE	99.69	99.51	99.69	**99.78**	99.60	99.51
HOSPITAL	97.64	**99.16**	98.36	96.98	95.58	96.42
PHONE	93.94	**98.55**	95.52	95.38	95.52	95.38
O	99.93	99.96	99.96	99.93	99.91	99.92

A.6.6 Learning Rate Performances with MEDDOCAN

Table A.7: Performance metrics for different entities across various learning rates on *Meddocan* dataset

Labels	5e-5	3e-5	2e-5	1e-5	5e-6
DOCTOR	**99.94**	99.88	99.82	99.88	99.88
PATIENT	99.94	99.94	99.94	99.94	99.87
DATE	**99.61**	99.46	99.46	99.38	99.38
HOSPITAL	**99.22**	98.44	96.88	97.67	91.97
STREET	**99.34**	99.30	99.30	99.22	99.06
COUNTRY	98.50	98.69	**99.06**	98.69	98.69
STATE	**98.75**	98.47	98.66	98.03	97.95
ORGANIZATION	75.73	68.75	84.11	80.73	84.11
ID_CONTACT	100.00	100.00	98.59	100.00	100.00
USERNAME	99.40	99.70	99.70	99.70	99.48
GENDER_PATIENT	98.99	99.21	98.99	**99.55**	98.77
FAMILY_PATIENT	**81.88**	80.00	81.33	81.82	78.71
AGE_PATIENT	99.11	**99.26**	99.11	99.16	98.96
EMAIL	77.42	77.42	77.42	77.42	77.42
PROFESSION	78.26	**90.91**	78.43	78.43	72.34
HEALTHPLAN	99.83	99.91	99.91	99.91	99.74

A.6.7 mBERT (Multilingual BERT):

[14] This model is designed to handle multilingual data and has been trained on a diverse set of languages.

```
tokenizer =
    BertTokenizerFast.from_pretrained("bert-base-multilingual-uncased")
model =
    BertForTokenClassification.from_pretrained("bert-base-multilingual-uncased",
    num_labels=num_labels)
```

A.6.8 sp (Spanish BERT - BETO):

[7] Specifically tailored for Spanish language data, this model leverages a Spanish-specific tokenizer and classification model.

```
tokenizer =
    AutoTokenizer.from_pretrained("dccuchile/bert-base-spanish-wwm-uncased")
model = AutoModelForTokenClassification.from_pretrained('dccuchile/
```

```
bert-base-spanish-wwm-uncased', num_labels=num_labels)
```

A.7 ILA

A.7.1 Database Columns

- **id**: A unique identifier for each record.
- **text**: The text content of the record.
- **manual_labels**: Labels manually annotated, serving as ground truth for training and validation. If a dataset is pre-annotated, real labels are used; otherwise, labels are initially set to 'O' for unannotated data.
- **predicted_labels**: Labels predicted by the model, which are displayed on the tool's web interface.
- **eval_records**: A flag indicating whether the record is used for training or evaluation, helping manage the distribution of data across different model training phases.
- **confidence**: Indicates the model's confidence in its predictions for each record, which is crucial for assessing model reliability.
- **manual_process**: A status flag that can take three values: 0 (not yet manually annotated), 1 (manually annotated but not yet used for training), or 2 (already integrated into model training).

A.7.2 Configuration File

```
{
    "project_name": "wikiNER",
    "num_records": 1072  0,
    "model_path": null,
    "tokenizer_path": null,
    "labels": [ "O", "PERSON", "LOCATION", "DATE","ID"],
    "label2id": {"O": 0, "PERSON": 1, "LOCATION": 2, "DATE": 3, "ID": 4},
    "id2label": {"0": "O", "1": "PERSON", "2": "LOCATION", "3": "DATE", "4":
   "ID"},
    "modelName": "roberta",
    "evalPercentage": 30,
    "trainingSteps": 10,
    "numPredictions": 50,
    "startFrom": 20
}
```

BIBLIOGRAPHY

[1] Tanbir Ahmed, Md Momin Al Aziz, and Noman Mohammed. "De-identification of electronic health record using neural network". In: *Scientific Reports* 10.1 (Oct. 29, 2020). Number: 1 Publisher: Nature Publishing Group, p. 18600. ISSN: 2045-2322. DOI: 10.1038/s41598-020-75544-1. URL: https://www.nature.com/articles/s41598-020-75544-1.

[2] Eiji Aramaki et al. "Automatic deidentification by using sentence features and label consistency". In: *i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data* (Jan. 1, 2006).

[3] Tom Arsenault. *SpanMarker for Multilingual Named Entity Recognition*. https://huggingface.co/tomaarsen/span-marker-mbert-base-multinerd. Accessed: 2024-05-18. 2023.

[4] David W. Bates. "Getting in Step: Electronic Health Records and their Role in Care Coordination". In: *Journal of General Internal Medicine* 25.3 (Mar. 1, 2010), pp. 174–176. ISSN: 1525-1497. DOI: 10.1007/s11606-010-1252-x. URL: https://doi.org/10.1007/s11606-010-1252-x.

[5] Bruce A. Beckwith et al. "Development and evaluation of an open source software tool for deidentification of pathology reports". In: *BMC Medical Informatics and Decision Making* 6.1 (Mar. 6, 2006), p. 12. ISSN: 1472-6947. DOI: 10.1186/1472-6947-6-12. URL: https://doi.org/10.1186/1472-6947-6-12.

[6] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: *EMNLP*. Association for Computational Linguistics, 2019. URL: https://www.aclweb.org/anthology/D19-1371.

[7] José Cañete et al. "Spanish Pre-Trained BERT Model and Evaluation Data". In: *PML4DC at ICLR 2020*. 2020.

[8] D. S. Carrell et al. "Is the Juice Worth the Squeeze? Costs and Benefits of Multiple Human Annotators for Clinical Text De-identification". In: *Methods of information in medicine* 55.4 (Aug. 5, 2016), pp. 356–364. ISSN: 0026-1270. DOI: 10.3414/ME15-01-0122. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5194214/.

[9] Pierre J Chambon et al. "Automated deidentification of radiology reports combining transformer and "hide in plain sight" rule-based methods". In: *Journal of the American Medical Informatics Association* 30.2 (Feb. 1, 2023), pp. 318–328. ISSN: 1527-974X. DOI: 10.1093/jamia/ocac219. URL: https://doi.org/10.1093/jamia/ocac219.

[10] Basit Chaudhry et al. "Systematic Review: Impact of Health Information Technology on Quality, Efficiency, and Costs of Medical Care". In: *Annals of Internal Medicine* 144.10 (May 16, 2006). Publisher: American College of Physicians, pp. 742–752. ISSN: 0003-4819. DOI: 10.7326/0003-4819-144-10-200605160-00125. URL: https://www.acpjournals.org/doi/full/10.7326/0003-4819-144-10-200605160-00125.

[11] Alexis Conneau et al. "Unsupervised Cross-lingual Representation Learning at Scale". In: *CoRR* abs/1911.02116 (2019). arXiv: 1911.02116. URL: http://arxiv.org/abs/1911.02116.

[12] Azad Dehghan et al. "Combining knowledge- and data-driven methods for de-identification of clinical narratives". In: *Journal of Biomedical Informatics*. Supplement: Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data 58 (Dec. 1, 2015), S53–S59. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2015.06.029. URL: https://www.sciencedirect.com/science/article/pii/S1532046415001392.

[13] Franck Dernoncourt et al. *De-identification of Patient Notes with Recurrent Neural Networks*. June 10, 2016. DOI: 10.48550/arXiv.1606.03475. arXiv: 1606.03475[cs,stat]. URL: http://arxiv.org/abs/1606.03475.

[14] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[15] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[16] Oscar Ferrández et al. "Evaluating current automatic de-identification methods with Veteran's health administration clinical documents". In: *BMC Medical Research Methodology* 12.1 (July 27, 2012), p. 109. ISSN: 1471-2288. DOI: 10.1186/1471-2288-12-109. URL: https://doi.org/10.1186/1471-2288-12-109.

[17] F. Jeff Friedlin and Clement J. McDonald. "A Software Tool for Removing Patient Identifying Information from Clinical Documents". In: *Journal of the American Medical Informatics Association : JAMIA* 15.5 (2008), pp. 601–610. ISSN: 1067-5027. DOI: 10.1197/jamia.M2702. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2528047/.

[18] Cyril Grouin and Aurélie Névéol. "De-identification of clinical notes in French: towards a protocol for reference corpus development". In: *Journal of Biomedical Informatics*. Special Issue on Informatics Methods in Medical Privacy 50 (Aug. 1, 2014), pp. 151–161. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2013.12.014. URL: https://www.sciencedirect.com/science/article/pii/S1532046413002050.

[19] Yu Gu et al. *Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing*. 2020. eprint: arXiv:2007.15779.

[20] Dilip Gupta, Melissa Saul, and John Gilbertson. "Evaluation of a Deidentification (De-Id) Software Engine to Share Pathology Reports and Clinical Documents for Research". In: *American Journal of Clinical Pathology* 121.2 (Feb. 2004), pp. 176–186. ISSN: 0002-9173, 1943-7722. DOI: 10.1309/E6K33GBPE5C27FYU. URL: https://academic.oup.com/ajcp/article-lookup/doi/10.1309/E6K33GBPE5C27FYU.

[21] Tzvika Hartman et al. "Customization scenarios for de-identification of clinical notes". In: *BMC Medical Informatics and Decision Making* 20.1 (Jan. 30, 2020), p. 14. ISSN: 1472-6947. DOI: 10.1186/s12911-020-1026-2. URL: https://doi.org/10.1186/s12911-020-1026-2.

[22] Dehuan Jiang et al. "A Deep Learning-Based System for the MEDDOCAN Task". In: *Deep Learning* (2019).

[23] Zhipeng Jiang et al. "De-identification of medical records using conditional random fields and long short-term memory networks". In: *Journal of Biomedical Informatics*. Supplement: A Natural Language Processing Challenge for Clinical Records: Research Domains Criteria (RDoC) for Psychiatry 75 (Nov. 1, 2017), S43–S53. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2017.10.003. URL: https://www.sciencedirect.com/science/article/pii/S1532046417302228.

[24] Alistair E. W. Johnson, Lucas Bulgarelli, and Tom J. Pollard. "Deidentification of free-text medical records using pre-trained bidirectional transformers". In: *Proceedings of the ACM Conference on Health, Inference, and Learning*. CHIL '20. New York, NY, USA: Association for Computing Machinery, Apr. 2, 2020, pp. 214–221. ISBN: 978-1-4503-7046-2. DOI: 10.1145/3368555.3384455. URL: https://dl.acm.org/doi/10.1145/3368555.3384455.

[25] Jitendra Jonnagaddala et al. "The OpenDeID corpus for patient de-identification". In: *Scientific Reports* 11.1 (Oct. 7, 2021). Publisher: Nature Publishing Group, p. 19973. ISSN: 2045-2322. DOI: 10.1038/s41598-021-99554-9. URL: https://www.nature.com/articles/s41598-021-99554-9.

[26] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *CoRR* abs/1909.11942 (2019). arXiv: 1909.11942. URL: http://arxiv.org/abs/1909.11942.

[27] Hee-Jin Lee et al. "A hybrid approach to automatic de-identification of psychiatric notes". In: *Journal of biomedical informatics* 75 Suppl (Nov. 2017), S19–S27. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2017.06.006. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5705430/.

[28] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4 (2020), pp. 1234–1240.

[29] Zengjian Liu et al. "Automatic de-identification of electronic medical records using token-level and character-level conditional random fields". In: *Journal of Biomedical Informatics*. Supplement: Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data 58 (Dec. 1, 2015), S47–S52. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2015.06.009. URL: https://www.sciencedirect.com/science/article/pii/S1532046415001197.

[30] Cedric Lothritz et al. "Evaluating Pretrained Transformer-based Models on the Task of Fine-Grained Named Entity Recognition". en. In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, 2020, pp. 3750–3760. DOI: 10.18653/v1/2020.coling-main.334. URL: https://www.aclweb.org/anthology/2020.coling-main.334.

[31] Jihang Mao and Wanli Liu. "Hadoken: a BERT-CRF Model for Medical Document Anonymization". In: (2019).

[32] Montserrat Marimon et al. "Automatic De-Identification of Medical Texts in Spanish: the MEDDOCAN Track, Corpus, Guidelines, Methods and Evaluation of Results". In: (2019).

[33] Christopher Meaney et al. *A Comparative Evaluation Of Transformer Models For De-Identification Of Clinical Text Data*. Mar. 25, 2022. DOI: 10.48550/arXiv.2204.07056. arXiv: 2204.07056[cs,stat]. URL: http://arxiv.org/abs/2204.07056.

[34] Stephane M. Meystre et al. "Automatic de-identification of textual documents in the electronic health record: a review of recent research". In: *BMC Medical Research Methodology* 10.1 (Aug. 2, 2010), p. 70. ISSN: 1471-2288. DOI: 10.1186/1471-2288-10-70. URL: https://doi.org/10.1186/1471-2288-10-70.

[35] Ishna Neamatullah et al. "Automated de-identification of free-text medical records". In: *BMC Medical Informatics and Decision Making* 8.1 (July 24, 2008), p. 32. ISSN: 1472-6947. DOI: 10.1186/1472-6947-8-32. URL: https://doi.org/10.1186/1472-6947-8-32.

[36] Bekelu Negash et al. "De-identification of Free Text Data containing Personal Health Information: A Scoping Review of Reviews". In: *International Journal of Population Data Science* 8.1 (Dec. 12, 2023). Number: 1. ISSN: 2399-4908. DOI: 10.23889/ijpds.v8i1.2153. URL: https://ijpds.org/article/view/2153.

[37] Joel Nothman et al. “Learning multilingual named entity recognition from Wikipedia”. In: *Artificial Intelligence* 194 (Jan. 2013), pp. 151–175. ISSN: 00043702. DOI: 10.1016/j.artint.2012.03.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S0004370212000276.

[38] Marco Polignano, Marco de Gemmis, and Giovanni Semeraro. “Comparing Transformer-based NER approaches for analysing textual medical diagnoses”. en. In: ().

[39] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *ArXiv* abs/1910.01108 (2019).

[40] Brett R. South et al. “Evaluating the effects of machine pre-annotation and an interactive annotation interface on manual de-identification of clinical text”. In: *Journal of Biomedical Informatics*. Special Issue on Informatics Methods in Medical Privacy 50 (Aug. 1, 2014), pp. 162–172. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2014.05.002. URL: https://www.sciencedirect.com/science/article/pii/S1532046414001191.

[41] Amber Stubbs, Christopher Kotfila, and Özlem Uzuner. “Automated systems for the de-identification of longitudinal clinical narratives: Overview of 2014 i2b2/UTHealth shared task Track 1”. In: *Journal of Biomedical Informatics*. Supplement: Proceedings of the 2014 i2b2/UTHealth Shared-Tasks and Workshop on Challenges in Natural Language Processing for Clinical Data 58 (Dec. 1, 2015), S11–S19. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2015.06.007. URL: https://www.sciencedirect.com/science/article/pii/S1532046415001173.

[42] György Szarvas, Richárd Farkas, and Róbert Busa-Fekete. “State-of-the-art anonymization of medical records using an iterative machine learning framework”. In: *Journal of the American Medical Informatics Association: JAMIA* 14.5 (2007), pp. 574–580. ISSN: 1067-5027. DOI: 10.1197/j.jamia.M2441.

[43] Arya Tafvizi and Maciej Pacula. “A Method for Automatic De-identification of Medical Records”. In: ().

[44] Buzhou Tang et al. “De-identification of Clinical Text via Bi-LSTM-CRF with Neural Language Models”. In: *AMIA Annual Symposium Proceedings* 2019 (Mar. 4, 2020), pp. 857–863. ISSN: 1942-597X. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7153082/.

[45] Simone Tedeschi and Roberto Navigli. “MultiNERD: A Multilingual, Multi-Genre and Fine-Grained Dataset for Named Entity Recognition (and Disambiguation)”. In: *Findings of the Association for Computational Linguistics: NAACL 2022*. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 801–812. DOI: 10.18653/v1/2022.findings-naacl.60. URL: https://aclanthology.org/2022.findings-naacl.60.

[46] Amina Tehseen et al. “Shahmukhi named entity recognition by using contextualized word embeddings”. In: *Expert Systems with Applications* 229 (Nov. 1, 2023), p. 120489. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2023.120489. URL: https://www.sciencedirect.com/science/article/pii/S0957417423009910.

[47] Özlem Uzuner, Yuan Luo, and Peter Szolovits. “Evaluating the State-of-the-Art in Automatic De-identification”. In: *Journal of the American Medical Informatics Association* 14.5 (Sept. 1, 2007), pp. 550–563. ISSN: 1067-5027. DOI: 10.1197/jamia.M2444. URL: https://doi.org/10.1197/jamia.M2444.

[48] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[49] Yunshun Zhong and Sebastian D. Goodfellow. “Domain-specific language models pre-trained on construction management systems corpora”. In: *Automation in Construction* 160 (Apr. 2024), p. 105316. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2024.105316. URL: https://www.sciencedirect.com/science/article/pii/S0926580524000529.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**