FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Detecting and Protecting Personally Identifiable Information through Machine Learning Techniques

Carlos Jorge Augusto Pereira da Silva

U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Mestrado em Engenharia de Software

Supervisor: Henrique Lopes Cardoso

July 27, 2020

Detecting and Protecting Personally Identifiable Information through Machine Learning Techniques

Carlos Jorge Augusto Pereira da Silva

Mestrado em Engenharia de Software

July 27, 2020

Abstract

This dissertation is a study on the automatic detection of Personal Identifiable Information (PII) in emails. The definition of PII has evolved following the technological developments and the misuse of PII.

PII storage has become easier, because of the diminishing costs and accelerated digitization of business. Nevertheless, the security aspects of data storage have been neglected resulting in an increasingly common availability of PII on the internet inadvertently taken from companies.

The digitization of business also contributed to the realization that there was value to be extracted from clients PII. Creating user profiles, adapting campaigns, and other techniques added value to their business, making companies willing to storage as much information as possible about their users, even if they had no use for it at the moment. For companies that had a software platform, there was the realization that the more data the customer inserted in the system the harder it would be for the customer to leave for a competitor. So, they made it difficult, or even impossible, to extract the information and move to another system, increasing customer retention rates. Each European country had their own legislation on data protection, making it more difficult for companies to run simultaneously in multiple European countries.

It is in this context that in 2018 the General Data Protection Regulation (GDPR) comes into force in the European Union (EU), looking into address these issues. Creating a single legislation across all European Union countries, promoting the data security best practices to protect consumers, and regulating the circulation of PII across the EU countries and companies in them.

An element that is common to most companies is the significant quantity of information that is stored and in circulation in emails. Including personal data of clients, employees, and collaborators. To follow the GDPR it is required an automated solution that allows the detection of PII at the high rate of email transactions.

In this dissertation we investigate how different models and Natural Language Processing techniques help to achieve that goal. The focus of these investigations is the design of a microservice that receives text content and detects PII named entities. For that, we trained machine learning models for PII detection and for segmenting emails into parts. We found that the state-of-the-art techniques are too expensive to run in production environments but that a good alternative could be achieved even if not achieving state-of-the-art results.

Keywords: Personal Identifiable Information, General Data Protection Regulation, Email, Natural Language Processing, Named Entity Recognition

Resumo

O objectivo da deteção de dados pessoais é a identificação e categorização de forma automática dos elementos relacionados com uma pessoa contidos num texto escrito em linguagem natural.

A definição de dados pessoais evoluiu acompanhando a evolução tecnológica e a má utilização de dados pessoais. O armazenamento de dados pessoais tornou-se mais fácil, pela diminuição dos custos a este associados e pela informatização das empresas. No entanto, foi negligenciada a segurança dos dados, tornando-se relativamente comum a publicação na internet de dados pessoais extraviados das empresas.

Por outro lado, com a crescente informatização das empresas - e com um progressivo aumento da informação armazenada sobre os clientes - aumentou a consciência da utilidade destes dados para definir perfis de utilizadores, adaptar campanhas, e outras estratégias que permitam uma vantagem comercial para as suas empresas. Isso levou a que as empresas retivessem toda a informação que obtiveram - mesmo que de momento não a estivessem a utilizar - na perspetiva de que no futuro poderia ser útil. Sabendo que a informação relativa a cada utilizador tem valor, não permitiam que o utilizador levasse essa informação para a competição, aumentando desta forma as taxas de retenção dos utilizadores.

A par disto, a legislação de cada país levantava um problema para a operação de uma empresa a nível europeu, tendo de se adaptar o seu funcionamento a cada país, segundo a legislação do mesmo.

É neste enquadramento que entra em vigor em 2018 o Regulamento Geral sobre a Proteção de Dados (RGPD) na União Europeia. Criando uma legislação comum a todos os países da União Europeia, promovendo as boas práticas de segurança de forma a proteger os consumidores e regulamentando a circulação de dados pessoais entre os países da União e empresas neles presentes.

Um elemento comum à maioria das empresas é a grande quantidade de informação que está armazenada e em circulação em emails. Incluindo dados pessoais dos clientes, colaborados e funcionários. Para cumprir com o RGPD é necessária uma solução que permita a deteção de dados pessoais de forma automática que consiga responder ao elevado volume de dados transacionados por email.

Nesta tese abordamos a utilização de modelos de aprendizagem máquina para a deteção de dados pessoais presentes em emails, em texto escrito em linguagem natural. Diferentes estratégias levam a equilíbrios distintos entre a eficácia e a eficiência do modelo. Esta tese explora diferentes modelos, baseados no estado-da-arte, e faz a sua comparação e avaliação relativamente à tarefa de deteção de dados pessoais em emails.

Keywords: Dados pessoais, Regulamento Geral de Proteção de Dados, Email, Processamento de linguagem natural,

Acknowledgements

I would like to thank my wife Ana for all the support, without whom I would have not been able to reach my goals towards completion of this degree while also balancing a busy career.

I would also like to show my gratitude to my supervisor, Professor Henrique Lopes Cardoso, and thank him for his guidance and patience.

Carlos da Silva

“The actual science of logic is conversant at present only with things either certain, impossible, or entirely doubtful, none of which (fortunately) we have to reason on. Therefore the true logic for this world is the calculus of Probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man’s mind.”

James Clerk Maxwell (1850)

Contents

List of Figures

List of Tables

Abbreviations

ANN	Artificial Neural Network
BiLSTM	Bidirectional Long Short-Term Memory
BPE	Byte-Pair-Encoding
BoW	Bag of Words
CNN	Convolution Neural Network
CRF	Conditional Random Field
DoB	Date of Birth
GDPR	General Data Protection Regulation
LSTM	Long Short Term Memory
NER	Named Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
OOV	Out-Of-Vocabulary
PHI	Public Health Information
PII	Personal Identifiable Information
POS	Part-of-Speech
Regex	Regular expression
RNN	Recurrent neural Network
WHO	World Health Organisation

Chapter 1

Introduction

Personally Identifiable Information (PII) is any information that can be used to identify a person. In particular, information that can identify a natural person directly, by referring to an identifier such as name or identification code, or indirectly by referring to one or more aspects specific to the person such as physical, physiological, mental, economic, cultural or social identifiers or descriptions. PII does not apply to legal persons such as companies and other corporations.

Sharing some PII with companies is important to our way of life. As we access health services, contact with lawyers, or open a bank account, we provide these organisations information they require to provide us with a service. We share that information by trusting that will not be used for purposes other than the ones to which we have agreed. Our bank account records can show that we regularly spend a significant amount of money in the liquor store. If the Bank sold that information to Insurance companies, then they could increase the premium or deny coverage. That would not be in our best interest and would break the trust contract with the Bank and the Insurance company.

We live in times of accelerated digitisation of our lives. The cost of storing information is decreasing. Moreover, companies are increasingly aware of the benefits of exploring vast datasets using data science. As more information is stored, and the capabilities for exploring that data are improved, the need to regulate and protect PII increases.

Different government regulations were set up to protect consumers and the market. Nevertheless, they did not all define PII in the same way. Companies must follow different regulations, depending on the countries they run their business and the business areas they work with. Doing business with military, the financial services or the healthcare brings different regulation to follow. If different countries have different regulations, and if by following the most comprehensive the other regulations are also followed, then complexity and costs can be removed by following the broadest regulation. The more comprehensive interpretation becomes the standard, as by following that one we ensure compliance with the remaining ones. The definition of PII used in the European Union (EU) General Data Protection Regulation (GDPR [60]) takes that position by defining PII as any information that can identify a person. The GDPR definition of PII is a very inclusive definition.

Written texts are the means by which many companies store PII. We can find them in emails, contracts and other legal texts, medical records, and so on.

The aim of Named Entity Recognition (NER) is the automatic processing of natural language text to detect and identify the named entities contained within it. The aim of PII detection, a subdomain of NER, is to identify the named entities that have personal information. This analysis receives the text as input. If the text contains PII, we aim to detect all the personal identifiers and return them as output.

Searching for PII in the text without limiting the type of document and the scope makes it a very challenging task. The different formatting of free text, the different writing styles and the subjectiveness of pieces of text are some of the main challenges. For example, on emails users tend to include signatures that include person names and contacts that would, in another context, be considered PII. That distinction is clear for humans visually inspecting the email content, but the automation of this skill is challenging.

In the ambit of GDPR, a person has the right to access the personal data collected about them, and the right to request that data to be erased. If PII information about a person is present in emails, that applies to those emails as well ([60, Article 15.e]). The company also must ensure that the PII of customer does not leave the company in an unauthorised way ([60, Article 4.12]). An employee that has access to customer PII cannot send that PII to outside the company without authorisation.

Compliance with data protection regulations in companies that process a considerable number of textual contents requires automated tools to do PII detection.

1.1 Motivation

GDPR [60] is in force since the 25th of May 2018. Non-compliance with the regulation risks administrative fines of up to 20 million EUR, or 4% of the total worldwide annual turnover of a transgressor company in the preceding financial year, whichever is higher. High fines motivate the adoption of international best practices like the ones documented in ISO 27001 [26], among others. One of the requirements of GDPR is to protect against unauthorised disclosure of, or access to, personal data (Article 5f) [60]. To comply with Article 5, PII has to be detected so that it can be protected [55].

In companies that deal with a considerable number of textual content, like emails, it is only possible to comply with the GDPR if the process can be automated and meet the production requirements. Traditionally, the detection of PII has been done using either a list of terms (gazetteer) or by writing regular expressions. The lists of all possible examples can get too big to manage manually. For example, the JRC-Names database of person names grows by 700 new names per week [25]. Comparing all email traffic of a company against these big databases can be very resource consuming.

In most cases, companies do not have these enormous lists of all possible values they want to match. In practice, no gazetteer could include all possible values. As such, they resort to regular expressions to identify named entities. The problem with regular expressions is that they grow in complexity as they are adapted to cover edge cases (instances that occur less often than the most common cases). Also, it becomes slower and more challenging to manage them [10].

Because of this, it is very interesting to investigate if, by using data science and machine learning, it is possible to automate the process of creating a model that would decrease maintenance overload and improve the quality of the results of PII detection [21].

Previous research in Named Entity Recognition had impressive results, motivating further investigations in the field. Nevertheless, previous research is, in most cases, still limited to well-formatted text or not focusing on PII [7, 5]. Research in PII detection is recent and accelerated by the increasing number of regulations in place.

1.2 Research Goals

This dissertation focuses on PII detection on email content. Email content is present in most of our work and personal life. It represents a log of our lives, what we buy, what we are interested in, with whom we contact, what hospitals, schools, restaurants we visit. Even more than our social profiles, as we expect the emails to be private. But while having such an ever-present in our lives, the number of datasets of emails available for research is minimal. Datasets are needed for developing and evaluating research. Publicly available datasets allow for different researchers to compare their solutions. That probably contributes to the limited number of recent research works that focus on email. PII is a scarce resource in datasets available for research, even more so now that the GDPR and similar regulations have been put in place.

Natural Language Processing (NLP) is “the set of methods for making human language accessible to computers” [23, p. 1]. Studying email content with the latest NLP developments, and connecting that with the recent focus on privacy regulations could not be more relevant.

The fundamental research questions that will be addressed in this thesis are the following:

1. Given an email, how to automatically detect and classify the possibly included PII named entities?

2. Can existing natural language processing machine learning architectures match corporate production requirements? The evaluation of the state-of-art architectures is done based on metrics such as F1-score, precision and recall, ignoring, in most cases, the other metrics relevant to production system such as model latency, number of requests per second during inference and the machine resources allocated.

3. Is it possible to learn, from annotated corpora, how to detect and classify PII named entities?

1.3 Main contributions

In this dissertation, we use supervised machine learning to detect PII on emails. That purpose implies the availability of a dataset of annotated PII named entities. To the best of our knowledge, no such corpus exists. In this dissertation, we introduce a corpus with labelled PII in English that we created. The most significant contributions of this dissertation are:

- computational models for PII detection in emails;
- an email corpus with annotated PII that can be used for PII detection evaluation and to support the development of new solutions;
- a solution for PII detection in emails that conforms with the pre-established requirements that are listed in Chapter 3.

1.4 Dissertation Outline

The dissertation is structured as follows: Chapter 2 presents the theoretical background relevant to PII detection. In Chapter 3 we list the production requirements for the PII detection systems, the libraries used, and the process used for the construction of the datasets. Chapter 4 describes the experiments we did and the evaluation of those. Chapter 5 presents the conclusions and points to directions of future work.

Chapter 2

Theoretical Background

The purpose of this chapter is to summarize the essential concepts and stages involved in PII detection.

2.1 Named Entities and Personal Identifiable Information

Named-Entity Recognition (NER) is an NLP task aimed at finding and classifying different named entity mentions in text. What constitutes a named entity type is task specific; *person*, *company* and *locations* are common, but it can be any other typified piece of information. The following example of Named Entity tagging is given in the proceedings of 1996 Message Understanding Conference [32], for the text:

> "We are striving to have a strong renewed creative partnership with Coca-Cola," Mr. Dooner says. However, odds of that happening are slim since word from Coke headquarters in Atlanta is that... (2.1)

The participants in the task would produce a report such as:

<ORGANIZATION-9402240133-5> :=
ORG_NAME: "Coca-Cola"
ORG_ALIAS: "Coke"
ORG_TYPE: COMPANY (2.2)
ORG_LOCALE: Atlanta CITY
ORG_COUNTRY: United States

PII is any named entity that can be used to identify a person. In the previous example, we have the name of a person ("Dooner") and his location ("Coke headquarters in Atlanta"), which are PII.

The majority of named entities are proper nouns [62], and that is the main reason why techniques like part-of-speech (POS) tagging are typically used in NE detection. However, not all text is appropriately formatted for those techniques to be precise.

The NER task has evolved from a small number of generic named entity types for English [75] to the exploration of different languages [78] and fine-grained ontologies [66]. Ontologies are a hierarchical organization that captures the subset/superset relations among words.

In 1995, Message Understanding Conference (MUC) [32] covered the retrieval of the following named entity types: names of *organisations*, *persons*, *geographic locations*, *dates*, *times*, and *quantities* (monetary values, percentages). The corpus was relatively small, with only 1129 named entity examples, and just for English. Of the named entities in MUC-6, person names are a type of named entity that is always considered PII. Organisation names can be PII if related to religion or political parties when we can connect that information to an individual as defined by Article 9 of GDPR [60]. At least in most cases, if we consider the phrase: "Pope Jorge Bergoglio is the head of the Catholic Church", considering "Catholic Church" as PII sounds wrong, but accordingly to GDPR it is correct. GDPR serves as clarification in situations where otherwise it would not be clear. Dates can be PII if they are dates of birth (DoB) and geographic locations can be PII if they correspond to the home addresses of people. The number of entities explored was limited, and only in English.

The Egunkaria 2000 corpus based on the news articles from the Basque newspaper Euskaldunon Egunkaria annotated with four categories: persons, organisations, locations and miscellaneous (those that do not belong in the previous three groups) [3]. The amount of text increased, it had the same limitations in terms of number of named entities, but the introduction of another language was very relevant to the field.

In 2002, the SIGNLL Conference on Computational Natural Language Learning (CoNLL 2002) shared task on NER was applied to text in Dutch and Spanish, promoting a language-independent approach to NER. The named entity types present in the task were the same four used in Egunkaria 2000. However, the number of named entity examples present in the CoNLL 2002 corpus is ten times bigger than Egunkaria 2000 just two years before. The number of entities continue to increase, as the number of languages. This allow to understand the specificities of each language and what are the techniques that can generalize. In 2003, the CoNLL shared task centred on detecting names of *persons*, *locations*, *organisations*, and *miscellaneous* entities, for English and German [78]. With 71045 named entity examples it was the biggest publicly available dataset of named entities to that point. Because of that, and the fact that it had an English version, it became the standard dataset to use for comparison of NER systems.

In Table 2.3, we identify the most relevant systems that have the best F_1 score on the CoNLL 2003 English shared task. From those 17 models, 10 use a Conditional Random Fields (CRFs), and 7 use a Bi-LSTM. The majority of recent developments in NLP do not report results in the NER task. The ones that do limit the results to the CoNLL 2003 task because the number of entities is limited, and the dataset is in English. In 2005 the BBN corpus [66] increased significantly the number of named entity records present. And in 2012, WikiNer presented 305460 annotated named entities. Other international conferences have also devoted attention to NER, such as ACE [22], INEX [19, 8], and TREC Entity Track [6]. There is a big overlap between the existing public available corpora. However, the field is lacking a recent corpus with entities where the existing state-of-the-art does not work well.

The NER task has been approached either with rule-systems, hybrid models or deep learning models. Rule systems use heuristics based on expert knowledge [55]. These were the first systems to appear in the 1990s. Hybrid models rely on domain-specific hints and linguistic analysis as features [76]. These systems learn from rules and humans [91]. Deep learning models use large datasets and extract the relevant features from them [63, 46].

Recently, the focus of research in NER is the use of deep learning methods using an annotated corpus of significant size. While there are many datasets for NER, the majority of the named entities they focus on are not PII.

2.1.1 NER dataset evaluation metrics

Each PII label the model predicts for the text can either be correct or wrong [39]. If it is correct it can be of one of two types:

- *True Positives* - correctly classified with that label
- *True Negatives* - correctly not classified with that label

Or if it wrong then it can also be of two types:

- *False Negatives* - the model should have detected the PII but did not
- *False Positives* - the model predicted a label when it should not have

2.1.1.1 Precision

The precision measurement is defined as:

$$precision = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2.3}$$

2.1.1.2 Recall

The recall measurement is defined as:

$$recall = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2.4}$$

The recall measurement is a measurement of the degree to which the process finds all the PII matches that are there.

In the ambit of GDPR, the person has the right to access the personal data collected about them, and the right to request that data to be erased. When we think about a system that is concerned about identifying PII for GDPR, recall is a very important measure. If a system has a low recall, that means that there will be much information about a person that will not be found, and the conformance of the system with GDPR is at risk.

2.1.1.3 F1 score

The F_1 score (2.5), the most common measure of the quality of a model, is the harmonic mean between precision and recall. This number is in the [0,1] range.

$$F_1 = 2.\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{2.5}$$

2.1.1.4 Accuracy

Accuracy is the degree to which the model correctly classifies all the possible true positives and only those.

$$accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives} \tag{2.6}$$

2.1.2 PII Types

Table 2.1 presents 30 types of named entities that can be considered PII, according to GDPR. Some of these named entity types are PII even if they appear in isolation (like the ID card number or the Tax File Number), while others require other types of information in the same context. Such is the case of disease or drug names. A disease name or a drug name do not identify anyone if they are not connected to other types of information that could allow such identification. Some PII, such as Telephone Number and Tax File Number, follow strict patterns, while other PII, such as Locations, do not.

Next, we analyse some of the named entity types that are relevant to PII detection in the context of GDPR:

- **Person**: The Person named entity refers to a name or nickname [44]. Even though names are usually considered public information, the correlation of a person name with other named entity types can make it possible to infer sensitive information.

- **Location**: As described by Barriere and Fouret [7], not all locations are considered PII. The birthplace locations or residence addresses are relevant. The birthplace can be used for discrimination based on ethnicity.

- **TIMEX**: Time and date references are named entities known as TIMEX [75]. Dates can be PII if they represent a date of birth, establishing age, or if linked to a record of a person movements as defined by Article 4 of GDPR [60].

Table 2.1: Named entities that can be considered PII

ALWAYS PII	CONTEXTUAL PII
Cookie identifiers	Date of Birth/Age
Criminal record (convictions and offences)	Disease names
Driver License	Drug identification codes
Email Address	Drug names
Employer Identification Number	Gender
ID Card Number	Health information
Medical records	Home address
Passport	IP address
Person Name	Locations
Social Security Number	Medical Procedure identification codes
Tax File Number (TFN)	Medical Procedure names
Trade union membership	Metadata
	Organisations
	Political opinion
	Race
	Religious beliefs
	Sexual orientation
	Telephone number

- **Personal Health Information** (PHI): PHI refers to any health-related information that can be used to identify a person, such as demographic information, medical history, insurance information, and other data that a healthcare provider uses to identify a person and determine appropriate care. With special attention to identifiers such as health plan beneficiary numbers and National Insurance Numbers, GDPR defines PHI as a special category of PII in Article 9, attributing special protections [60]. Because in the recent past in Europe that information was used to discriminate and kill Europeans. Some of the PHI types have a format that is very specific to the sender/recipient, such as in the case of hospital account numbers where each hospital can have a different format.

To the best of our knowledge, there is no dataset for PII, that is publicly available. Although some elements of some public datasets can be considered PII, they are not useful to us because they are based on formal texts like news articles. Our focus in this dissertation is on free text, specifically emails. Some email datasets are available (such as Enron email dataset [43]) but the PII present in them is not annotated, and, in most cases, those datasets are old enough to no longer represent the current practices in email exchanges.

2.1.3 Challenges in detecting PII

There are many challenges to overcome in the detection of PII. Multiple languages can be present in the same text making language detection harder [81]. The difficulty to find examples of certain types of PII like national health identification numbers (unbalanced data [68]) make the training of model harder. One of the strategies explored in commercial tools like prodigy[1] to deal with unbalanced is to use language models to create gazetteers from a small number of examples. Then, in a process of active learning, a dataset can be quickly annotated.

Free text, like emails, differ from formal texts, like newspaper articles, in many aspects such as form, length, type of language used, number of misspellings, between others [54, 51, 18, 7]. Out-Of-Vocabulary (OOV) words are problematic, but in the case of misspellings, they can be generated and used as a feature to improve the model [65].

As mentioned in [54] the emails have some structured information, like the email headers, that contain person names, we do not consider these to be PII since they are the names of the persons involved in the conversation.

Signoffs and disclaimers are challenging since they are not related to the main message. For example, if a person has an email signature with their phone number and email address that would not be relevant to GDPR since the person is supplying the information.

Another challenging situation would be the mixture of PII from one country in a text of an unrelated language (i.e. an email written in English with PII of a Portuguese person). As most models are language dependent and embedding, they would fail the PII detection in that case.

Some PII types present specific challenges:

- **Person**: The spelling of a person name can vary because of misspells [25], making its detection using gazetteers harder. Some people names have other meanings, like names of months (Theresa May), flowers (Daisy Lowe) or common objects (John Books).

- **Location**: It is difficult to identify the addresses that are PII from the other types of Addresses that are not. This type of PII detection is a good example where gazetteers fail – since a gazetteer is simply a list of words, it does not allow us to distinguish any location from a personal Address. Addresses can be written in various formats, and this presents an added challenge for Address detection.

- **TIMEX**: There are several formats for TIMEX representation making the task of detecting them harder. For instance, 85 TIMEX formats are listed in ISO 8601 [27], with an explanation of the elements that compose those date and time representations. Having strict rules and patterns the TIMEX formats look, at first, as a good fit to be detected using Regular expressions (regex) as shown by Neamatullah [55].

  However, some patterns are not unique enough to determine if they are a date or not, leading to a high number of false positives if we rely just on regex to detect them.

  For example, let us consider the following piece of text:

$$\text{Order \#A 19850412 16 Placed on May 08, 2020.} \tag{2.7}$$

[1] https://prodi.gy

19850412 can be a representation of 4th December 1985 so a regex would wrongly match it as a TIMEX:

$$\text{Order \#A } [19850412]_{TIMEX} \text{ 16 Placed on } [May08, 2020]_{TIMEX}. \tag{2.8}$$

Collocations, concordance, and term frequency are useful techniques to detect elements like "last year" and "next year" as a TIMEX named entity type.

- **Personal Health Information** (PHI): One type of PHI is the International Classification of Diseases (ICD), an alphanumeric representation of diseases, other health problems and clinical procedures. Although ICD is very relevant to PII detection, since it represents health information, it can be problematic to identify. One example of an ICD is "A31.0" that represents a Pulmonary mycobacterial Infection. That alphanumeric sequence can appear in different formats ("A310", "A31-0", "A31 0", "a31 0", ...). Furthermore, it can appear as part of a message in another context, such as a reference to an airplane type (Airbus A310) or as a postcode. The context will be essential to decrease false positives by making a distinction between those two situations.

It would be hard to train a model to handle these characteristics, that are specific to email content, on a dataset of another type of text.

These PII have to be annotated into a corpus, and then that information needs to be preprocessed and transformed into a format that is suitable to feed deep learning models. Those are the tasks of text normalization, tokenization and encoding that we are going to describe next.

2.2 Data Pre-processing and Representation

From a raw format, the data needs to be transformed into a set of representations suitable to train the model. First, we need to clean data because any model based on inaccurate data can produce misleading results. We want to ensure that the data we work with is quality data. Secondly, we need to transform the alphanumeric content into a numeric representation. All machine learning models operate on numeric data, so we need to map the text into numbers.

2.2.1 Text normalization

Text normalization is a process of reducing the vocabulary size by removing or substituting tokens, resulting in smaller vocabulary, but also affecting the performance of the NER model. By having a smaller vocabulary, text normalization reduces the memory needed to run the model and increases the speed of prediction [23]. It reduces the number of OOV words and helps generalize the model by, in most cases, treating in the same way tokens whose distinction does not have semantic value. The critical part is to limit the normalization to the differences that do not have semantic value for the task. For example, for the task of PII detection, the casing of words is a useful feature, but a common normalization step is to lowercase all text, losing that information. Normalization is

Table 2.2: Examples of normalization

Type	**Example**
Contractions	“Didn’t” to “Did not”
Different spelling	“analogue” to “analog”
Lemmatization	“ran” to “run”
Lower casing	“May” to “may”
Stemming	“consistently” to “consist”

dependent on the task that the models have to do. For NER, the case of the characters has semantic value and helps make the model more accurate [74], so lower casing the text would be a mistake.

Some examples of normalization are shown in the Table 2.2. Removing contractions helps to reduce the number of out-of-vocabulary by transforming, for example, all instances of “don’t” changed into “do not”. Changing the words that have different spelling in different countries and lower casing all text are two other techniques used for the same reason.

2.2.2 Feature representation

To train a machine learning model, we need to transform the text into a form that is useful for the model. One of the most straightforward ways we have to encode tokens is to use a technique called Bag of words (BoW) [53]. Bag of words gets its name from mathematics, where a bag is a set, but that might contain duplicates. Order is not kept [79]. We start with a vocabulary V of all the possible words in the vocabulary. Then, if we want to represent a sentence for each word x, we count the number of times it appears in the sentence. In the end, we have a set, for example:

$$y = [0,1,0,2,4,4,2,1,0,0] \tag{2.9}$$

where y_x is the count of the word x in that sentence. The length of y is equal to by definition to the euclidean length of the vector V

$$V \triangleq |V| \tag{2.10}$$

For example, if we have a vocabulary of the words:

$$[big, brown, cat, cow, dog, fox, jumps, lazy, over, quick, the]. \tag{2.11}$$

For a sentence (2.12), we count the number of times each word occurs:

$$\text{The lazy dog jumps over the quick brown fox.} \tag{2.12}$$

$$[\ 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 2\ ] \tag{2.13}$$

Then another sentence comes along, and we continue to do the same process:

$$\text{The lazy cat jumps over the big brown cow.} \tag{2.14}$$

$$[\ 1, 2, 1, 1, 1, 1, 2, 2, 2, 1, 4\ ] \tag{2.15}$$

The resulting vector (2.15) represents both sentences of our text, retaining the information of how often each word appears in the text. However, it does not retain any information about how the order in which the words appear. BoW is the basis for algorithms like term frequency-inverse document frequency (TFIDF) that is extensively used in the information retrieval field. The concept of bag-of-words is straightforward but yet effective for text classification. It is used as a benchmark for sentence classification [38]. And, relevant in state-of-art solutions like the "Semantic Product Search" from Amazon [57].

However, BoW is rarely used in NER systems these days, in favour of recent representation techniques. One of the alternatives to BoW representations is token embeddings. Token embeddings are the term by which we know the technique of representing tokens as n-dimensions vectors. The tokens can be characters, parts of words, words, sentences or documents, depending on the task. Each type of embeddings has its strengths and weaknesses:

- **Character embeddings** are useful to make the model more resilient to OOV (OOV decrease the model performance), including misspellings, prefixes and endings not represented in the dataset [47]. They have the benefit of taking less space, helping to keep the models small. However, they do not capture the same relation between words as word embeddings do.

- **Sub-word embeddings**, also known as wordpieces, are the embeddings of the division of words into a limited set of common sub-word units. They provide a balance between the character embeddings and the words embeddings [11]. One of the most relevant implementations is the Byte-Pair Encoding (BPE) [71].

- **Word embeddings** are a relevant encoding technique to capture word meaning with contextual information [85, 4]. They have shown compelling results in capturing the relation between words based on the context [90]. They are known to be less resilient to OOV than character and sub-word embeddings. Word embeddings show lower performance in content that is poorly formatted [65].

- **Sentence embeddings** are vector representations of phrases or sentences. They have been shown useful in a variety of tasks, including classification [89] and ranking [59]. Sentence embeddings have improved NER systems by better representing sentence patterns [86].

2.2.3 Tokenization

Tokenization is the process of dividing text from a sequence of characters to a sequence of tokens. We obtain these tokens by using a simple approach like defining the subset of characters representing whitespace and then splitting on these tokens. However, more advanced strategies appeared to reduce vocabulary size and to improve the handling of OOV words, such as character [41] and sub-word tokenizations [84].

Choosing the right tokenization library depends on the task, the performance, the human languages it is valid for, the licensing, if the library is maintained, if the software code can be inspected, between other aspects.

The tokenization technique to be used can also be influenced by the pre-trained embedding to use. Different embeddings can use different tokenization techniques, and to benefit from them the model has to use the same tokenization strategy that was used for creating the embeddings. In some cases, the same text is tokenized more than once, for example, to create character embeddings and words embeddings from the same text. Previous works have shown that using different embeddings together can improve the quality of the model [11, 4].

2.3 Deep Learning Models

For a long time, the main NLP techniques used were machine learning approaches such as support vector machines or logistic regression. Those were trained over very high dimensional yet very sparse feature vectors. Part of the reason for that was the limited computational resources available. The models were smaller and relied heavily on manual annotation work. Recently, the field has moved from such linear models over sparse inputs to non-linear neural network models over dense inputs. That has improved metric scores, although, in general, these models take more resources to train and use. In this section, we will review some of these models and their usage on NER tasks.

2.3.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a category of neural networks that accepts input data in a matrix format where a convolution operation can be applied. Each convolution layer filter represents a feature of interest from the previous layer. CNNs are usually seen as only looking at position-invariant features of their inputs [46]. The two main ways CNNs are used in NER are:

- to extract a fixed-length feature vector from character-level features [77, 12, 50]. Previous studies ([12, 83]) have shown that CNN is a useful approach to extract morphological information (such as the prefix or suffix of a word) from characters of words and encode it into neural representations.

- as the main neural network architecture [92, 15]. Neural network approaches to NER mainly follow from the work of Collobert et al. [15], who applied a CRF on top of a CNN.

2.3.2 Recurrent Neural Network

A Recurrent Neural Network (RNN) [24] is an architecture that, as it processes the input, it can remember what it has processed ins sequence, it has a memory. It was meant to deal with temporal data. NER can be described as a sequence to sequence problem, where the input is a sequence of words, and the output is a sequence of named entity tags that match the input words. This concept perfectly fits the RNN architecture. The two most relevant RNNs at the moment for NLP are Long Short-Term Memory (LSTM) and Bidirectional Long Short-Term Memory (Bi-LSTM).

The RNN proposed by Elman [24] can learn long dependencies, but as demonstrated by Hochreiter [35] they suffer the *vanishing gradient problem*. To overcome that problem, Hochreiter and Schmidhuber proposed Long Short-Term Memory (LSTM) [36] model.

A Long Short-Term Memory (LSTM) network [36] is a type of RNN that can learn long dependencies, usually from previous words. Previous work has found that LSTM networks use around 200 tokens as context [40]. LSTM networks have obtained reliable results on multiple benchmarks, including CoNLL 2003, as we can see in Table 2.3.

The main components of an LSTM are: a memory cell or cell state, an input gate, an output gate and a forget gate. At each step, the LSTM decides what information is no longer relevant and can discard, by using the forget gate. Irrelevant information is removed from the cell state. Next is to decide what new information to add to the new cell state. In the input gate, a decision is made as to what values of the cell state to update. Finally, the output gate is used to decide what is expressed by the cell. LSTMs can maintain longer dependencies than Elman RNNs by making use of gates to selectively keep relevant dependencies and discard the ones that are not.

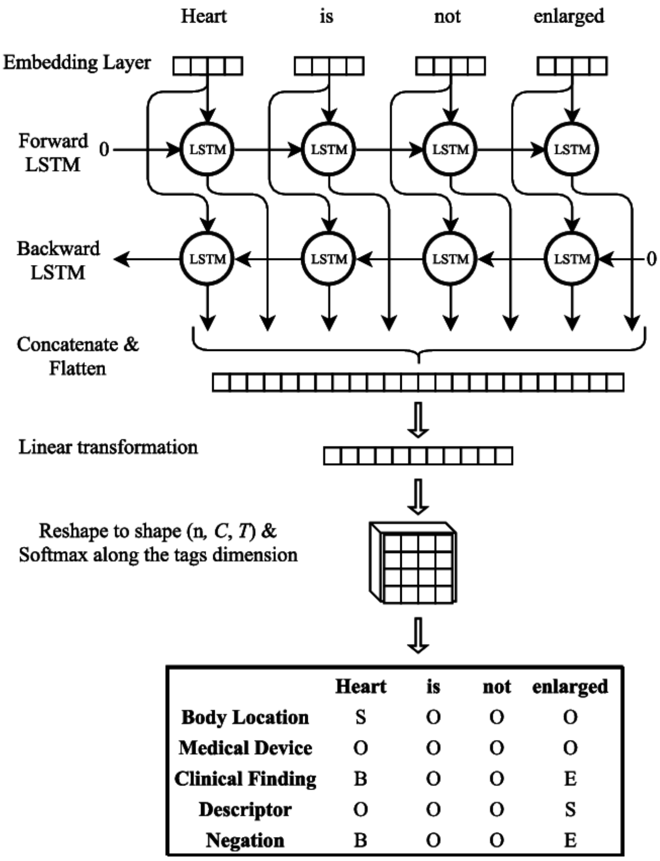
Even though in Table 2.3, we see several references to LSTM, one should keep in mind that not all of them share the same configuration. One of these modifications that gain attraction is the Gated Recurrent Unit (GRU) [13] used in the TagLM model [64].

The main problem with LSTM is that the logic to compute the relationships inside the sequence can not be computed in parallel.

A Bidirectional Long Short-Term Memory (Bi-LSTM) architecture [31] is an LSTM that not only gets information about previous words, but that also gets information from the words that follow. A Bi-LSTM is achieved by having two LSTMs, as illustrated in Figure 2.1, one that processes the text in one direction (forward LSTM) and the other that processes it in the other direction (backward LSTM). When we merge the representation of the words from both LSTMs, they produce better results than one individual LSTM.

Bi-LSTM achieves better results on PII detection as we can see in Table 2.3, because, in many cases, the meaning of a word also depends on the words that come next in the sentence (i.e. “Teddy Bear” vs “Teddy Roosevelt”). Bi-LSTMs were used in many state-of-the-art NER systems [77, 46, 86, 73, 4]

The output of Bi-LSTM is useful for a task of detecting PII but it lacks a way of representing the heuristics that are essential to the definition of many PII types. One way of learning those rules is to feed a Conditional Random Field (CRF) [45]. This is what was proposed by



	Heart	**is**	**not**	**enlarged**
Body Location	S	O	O	O
Medical Device	O	O	O	O
Clinical Finding	B	O	O	E
Descriptor	O	O	O	S
Negation	B	O	O	E

Figure 2.1: Bi-LSTM memory block [16]

Lample *et al.* [46], and the structure used by the state-of-the-art solution [5]. The CRF will automatically learn the heuristics from the labelled data, learning that the probability of a word being tagged with a given label depends on the assigned tag to the word before.

2.4 Language Models

Language models are machine learning models that calculate the probability of a word, or sequence of words, in a certain context of neighbouring words. All the best performing models for NER use language models [77, 73, 4, 20], so next we will review some of them that were used for NER by chronological order:

1. Elmo: Peters *et al.* [64] show the value of contextualised word embeddings achieving state-of-the-art F1 score at the time, work that was later improved in the creation of ELMo [63]. ELMo uses character-based convolution filters that are then passed through two LSTM layers to create a word representation that depends on the context and is resilient to words not seen during training.

   In contrast with the classic word representations, uses the entire context in which a word appears to form its word representations. Each word has multiple embeddings presentations depending on the context where they appear. For example, if we consider these two word-sequences: “How may I help you?” and “Let’s schedule that to 12 may.” the word “may” will have two different representations one for each context where it has appeared. This results in the two instances of the word “may” to be treated as two different words.

2. BERT: BERT stands for “Bidirectional Encoder Representations from Transformers” [20]. It picks up the idea from the work of ELMo - that we should train a neural network on a significant corpus in an unsupervised way and use that to several NLP tasks. They have pushed to the next level by proving that the results are improved by adding self-attention from Transformer [80] and contextual information by creating a bidirectional Transformer. Instead of using character-based embeddings like ELMo, it uses the sub-word embeddings Byte-Pair Encoding [71].

3. RoBERTa: RoBERTa (A Robustly Optimized BERT Pretraining Approach) [49] replicates the BERT system with new values for the hyperparameters, more data, and trained with more computing power to achieve better results than BERT in many NLP tasks. It does show how important is the choice of values of the hyperparameters.

4. DistilBERT: DistilBERT learns a distilled (approximate) version of BERT, retaining 97% performance but using only half the number of parameters and being 60% faster. DistilBERT uses a technique called knowledge distillation, where a smaller model (DistilBERT) is trained to reproduce the larger neural network (BERT).

   The idea is that the BERT model did learn a smaller part of logic that is responsible for most of the good performance of the model, and a larger part of rules that have a much smaller impact. So DistilBERT learns from BERT directly instead of learning from the training data used to train BERT.

2.5 State-of-the-Art

In 2003, the CoNLL (CoNLL 2003) shared task centred on detecting names of *persons*, *locations*, *organisations* and *miscellaneous* entities (that do not belong to the previous three groups), for English and German [78].

In Table 2.3, we identify the most relevant systems that have the best F_1 score on the CoNLL 2003 English shared task. From those 17 models, 10 use a Conditional Random Fields (CRFs), and 7 use a Bi-LSTM. The majority of recent developments in NLP do not report results in the NER task. The ones that do limit the results to the CoNLL 2003 task because the number of entities is limited, and the dataset is in English.

Table 2.3: CoNLL 2003 (NER in English)

Model Name	***Technique***	F_1
Cloze	Bi-LSTM-CRF-CNN Large+fine-tune [5]	**93.5**
	Bi-LSTM-CRF+ELMo+BERT+Flair [73]	93.38
Flair	Bi-LSTM+Flair embeddings [4]	93.09
BERT Large	BERT Large+Bi-LSTM [20]	92.80
CVT	Bi-LSTM + Multi-Task [14]	92.61
BERT Base	BERT Base [20]	92.4
ELMo	Bi-LSTM-CRF+LM [63]	92.22
TagLM	LM+embeddings [64]	91.93
Neural-CRF-AE	Bi-LSTM-CRF+AutoEncoder+ Lexical Features [83]	91.89
	Bi-LSTM-CRF+Lexical Features [29]	91.73
LM-LSTM-CRF	LM-LSTM-CRF [48]	91.71
	Bi-LSTM-CNN [12]	91.62
	Hybrid semi-Markov CRF [88]	91.38
	IXA pipes [2]	91.36
NCRF++	CCNN+WLSTM+CRF [85]	91.35
	CRF-GRU [87]	91.26
	Bi-LSTM-CNNs-CRF [50]	91.21
	Bi-LSTM-CRF [46]	90.94
GloVe	GloVe [61]	88.30
CBOW	CBOW [52]	88.20*

*value from [61]

The best result on CoNLL 2003 (the one by Baevski *et al.* [5]) uses a Convolutional Neural Network (CNN) with 330 million parameters and applies a Bi-LSTM-CRF based on the work of Peters et al. [63] and Huang et al. [37]. The model uses a masking technique, similar to the one used by BERT [20], where the words surrounding the target word are used to predict the probability of the word appearing in that context. The masking technique has initially been proposed by Wilson Taylor in 1953 as the “Cloze” task [77], which is the reason for this model name. The words are encoded using character embeddings that are passed to convolution layers based on the work of Kim et al. [41], and very similar to what is done in ELMo. It is also relevant that the authors did try BPE and found that the character CNN gave them better results.

In the second place comes the work from Straková et al. [73] combining multiple pre-trained embeddings (ELMo, BERT and FLAIR) with their embeddings that include forms, lemmas and POS tags information. While it achieves impressive results, it is a model that depends on more pre-processing, and that is heavier than the Cloze model.

The Flair model [4] has the third-best result on CoNLL 2003. It introduces a new type of embeddings combining characters embeddings and contextual characters. At a higher level, it looks very similar to the ELMo model, as both use contextualized character embeddings. However, while ELMo passes the characters through convolutional filters and then to the LSTM layers, the Flair model passes the characters directly to the LSTM layers. From the work of Straková et al. [73] we deduce that even though they look very similar, ELMo and Flair models capture different and complementary information, as it measured that using the two models together did achieve better results than either of the two embeddings used alone. To note that Flair system [4] refers that character embeddings by themselves got a worse result than the ELMo model, and only by combining Flair with Glove embeddings, it got better results than ELMo. We have no information if Flair + Glove embeddings achieve better results than ELMo + Glove embeddings or not. All we can conclude is that character and word-level information put together is helpful to achieve better performance.

2.6 Conclusion

ELMo represented a significant shift in NLP, putting languages models at the heart of current NLP practice. Each model after it represented slightly better results in different metrics, from Facebook’s RoBERTa that presents higher F1-score but at the cost of speed, to DistilBERT that compromises between speed and accuracy. RoBERTa and DistilBERT represent symbolically two of the current trends in language models, go bigger and achieve better results, or go smaller and try to approximate BERT performance.

In the papers that constitute the state-of-the-art for NER, we see a prevalence of Recurring Neural Networks, in particular, Bidirectional LSTMs. Character embeddings with contextual information are the base idea behind Cloze, Flair and ELMo models, and produce good results. Nevertheless, using them in combination with word embeddings produced better results in some cases.

Chapter 3

Methodology

This chapter details the requirements and guidelines which govern the implementation of the system. It also enumerates the libraries used and the process of data preparation. To build the PII in email detection system, we need a set of guidelines. The requirements will be specific to the email domain and the production environment where the system will be deployed. The process of preparing the data for training the model can be divided into several phases: data collection, pre-processing, annotation, and review, as shown in Figure 3.1.

3.1 Requirements

The system is based on the recognition and classification of PII content in written data sources, specifically emails. Considering the high volume of email content that needs to be analysed while in traffic, a set of requirements was identified to guide our system towards this task. A PII in email detection system is only useful if it can fulfil the software requirements defined for it. The requirements in Table 3.1 guide the decision process while developing the PII detection system. There is a limited amount of work done, published and publicly available on how to use machine learning models in production systems, the decision process about the algorithms to use and the tradeoffs it entails. The system is intended to replace an existing system that does PII detection mainly using regular expressions and similar techniques. The REQ-6 (Table 3.1) defines a metric for how big the model should be, that is defined because in a cloud infrastructure and agile environment the model could be updated frequently. If the model size is too big it will cause delays. If the model is being deployed to several hundred machines at the same time the process will take longer and increase operational response times.

Table 3.1: Requirements

	Description
REQ-1	The system must accept inputs of different sizes, as email body sizes vary. The minimal size being one character. The max size being 20 MB of text.
REQ-2	Given the same input the system must give the same output.
REQ-3	The system should allow setting up a list of terms to be ignored if detected so that I can deal with any exceptional case that requires a bypass.
REQ-5	The system must outperform the F1-scores of the previous system for the named entities that are already covered by the old system.
REQ-6	The size of the model should not exceed the 500 MB.
PER-1	The system must present the result in less than five seconds for 95 percent of requests, running on CPU.
PER-2	The system must be able to respond to multiple threads at the same time.
PER-3	The system must only use open source libraries that have a license that allows commercial use.
REL-1	No more than five experimental runs out of 1000 can be lost because of software failures.

REQ: Functional requirements; PER: Performance requirements; REL: Reliability requirements

3.2 Libraries

We used ICU4J to normalise the text, reducing the set of Unicode code present. The ICU library is widely used for text normalisation.

Data annotation is a very time-consuming task, and to speed it up, we used Prodigy. We have tested many annotation tools and ended up with Prodigy because it allowed us to achieve the fastest pace of annotation. To annotate data in Prodigy, we had to transform all content into a JSONL format, where each line represented an email.

We have used Python for most of the project as it is the programming language most widely used for machine learning and natural language processing.

We have utilised the spaCy library, one of the most prominent libraries in the NLP community, written in Python and with a compelling performance both in terms of speed and memory usage. Many of the models in Table 2.3 use it. We used it for tokenization, for the division of text content into sentences and part-of-speech analysis. We have also used their CNN for training a model.

The HuggingFace transformers [82] presented an easy way to experiment with different language models. Since it is written in Python, it is easy to integrate with spaCy and the other libraries we were using.

3.3 Datasets

The process of preparing the data for training the model can be divided into several phases: data collection, pre-processing, annotation, and review, as shown in Figure 3.1. It is an iterative process in which we will acquire additional knowledge about the domain and refine our approach. As an overview of the process:

1. In data collection we gather the data that will constitute our dataset, from Enron dataset, personal emails and newsgroups.

2. The data cleaning step, also known as pre-processing, is where we the data is transformed into an adequate form for annotation.

3. In the annotation step a person examines the data and labels it accordingly with the objective of the process.

4. The learning-curve evaluation is a measurement if by having more that the accuracy of the model would increase. If it shows that more data is needed the process iterates back to data collection step.

5. Once the amount of data is deemed sufficient, the process goes to the data review step, were all the data is analysed looking for mistakes and inconsistencies in the annotations. If errors of data cleaning are found the process resets to the data cleaning step. If the review signals issues in data annotation the process resets to that step.

6. If no issues are found the dataset is ready to be used in training a machine learning model.

We detail each phase of the data preparation process in the following sections.

3.3.1 Data collection - Raw Data

The existing datasets for Named Entity Recognition are based on either news articles or Wikipedia pages.

We can divide text content into two types: Formal and informal texts. Formal text is usually longer, has limited use of slang and colloquialisms, misspellings are uncommon. Examples of formal text are legal documents, newspaper articles and Wikipedia pages. On the other side, informal texts are written in a more casual style, usually shorter, and where slang, colloquialisms, and misspellings are more common. Examples of the informal text include email messages, twitter posts and Reddit posts.

The differences between the two types of text affect NLP analysis. For instance, the part-of-speech is harder in informal text because authors are less careful with text structure and misspellings than in formal texts [7]. Misspellings are, in most cases, out-of-vocabulary words, and that diminishes the effectiveness of the use the embeddings.

Formal texts like news articles are different from informal texts like email content. News articles follow strict rules in the way that they are written, and are reviewed before publishing. A similar review process does happen on Wikipedia articles. Those types of text are different from email content, where a review process is rarely done. For both those cases, the number of

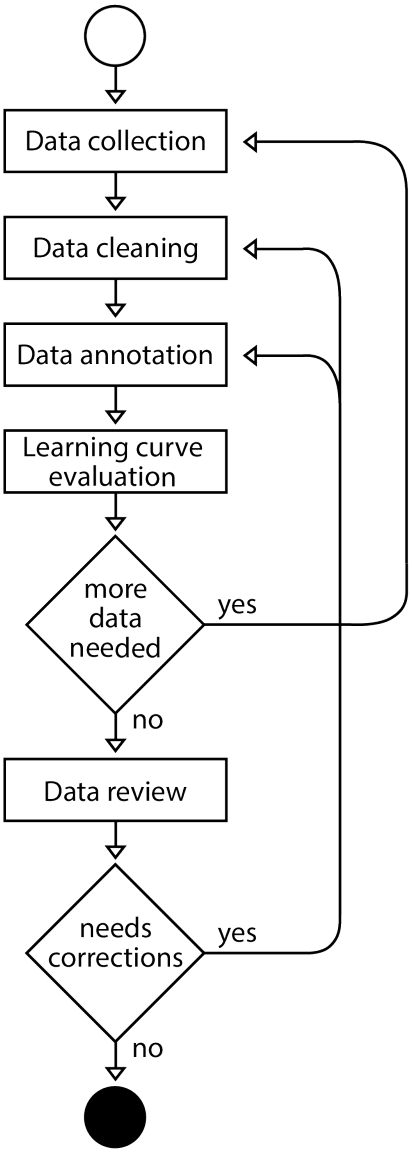


Figure 3.1: Data preparation process workflow

misspellings, for example, is much lower than what we expect in email content. News-articles tend to use refined vocabulary, that is not common in emails.

With these differences the existing datasets (trained on formal texts) are not an adequate representation of the world where we want to apply the PII detection system (informal text). That is not to say that some emails will not have formal text in them, but they are a minority. Another characteristic of the existing datasets is the length of the text, that is typically longer than email content.

For those reasons, we had to construct our dataset for PII detection from multiples sources, including Enron data, newsgroups, and personal emails to add more recent content to the dataset.

3.3.2 Data cleaning - Pre-processing

Before annotating the dataset, we need to clean it by removing elements that might make the annotation and the machine learning process more difficult. One of the tasks of pre-processing is normalization.

Normalization is the task of converting text into a more standard form. For example, it consists in transforming texts in different Unicode to the same Unicode. Since the emails come from various sources, they come with many differences in form, so by changing them to a standardised form this step simplifies the work of the next tasks. We used ICU4J to normalize the text, converting all text to UTF-8, removed control characters and normalized all the different representations of space to a single one. The capitalization of words has a semantic value that is useful for Named Entity Recognition tasks [74], so the tokenization process maintained the casing of the words.

3.4 Annotated data

We annotated two different datasets, one for the email segmentation and another for PII detection. In the next sections we will look into each one of those datasets in more detail and how they were annotated.

3.4.1 PII detection dataset annotation

To develop a model for PII detection, we need to have a useful dataset with annotated PII. One person annotated the emails for four months. The annotator is not a native speaker of English.

The dataset contains examples of emails where the words are labelled for what the entity class is. From those annotations, the machine learning algorithms generate models to address some specific task automatically. Therefore, producing a collection of annotated documents (corpus) with PII is a fundamental requirement. For PII detection, we have annotated 10473 emails. In Table 3.2 we show the counts per PII entity type. The dataset is very unbalanced, as the presence of some PII types is much more frequent than others in emails. The most frequent class has a representation of 34% (TIMEX), while the one with the lowest representation (IP_ADDRESS) stays at 0.01%.

Table 3.2: Annotated PII counts

Named Entity	Number of annotated elements
IP_ADDRESS	15
PASSWORD	82
USERNAME	156
DRUG	1982
TITLE	1986
ADDRESS	3510
EMAIL_ADDRESS	4811
GPE	6441
PHONE_NUMBER	10920
ORG	19363
PERSON	50120
TIMEX	51394

3.4.2 Email segmentation dataset annotation

Since we consider an email segmentation model useful in processing the email, we have also created an annotated dataset for that task.

There were three classes to be identified in each email:

1. Reply: any part of an earlier email that this message was replying to;

2. Signature: that includes all the information at the end of the email, including signatures and disclaimers;

3. Other: the rest of the email that is not one of the previous categories, the main message of the email.

The disclaimers present at the end of the email can have addresses, names of companies and persons, but they are not relevant to PII detection. By detecting the disclaimer section, we can ignore it. In Table 3.3 we how an example of an email annotated for email segmentation. In that example we can see that a disclaimer taking 12 lines out of 31 email lines. Disclaimer are not present in all emails, but in ones where they are present they can represent a significant part of email.

The email signature usually contains at least a person name, including sometimes the title, but can also include contact information like email address, phone numbers, address, or organization. We have not made a distinction between the email signature and the disclaimer in the classification, we have treated them as SIGNATURE. The reason for that is that the difference is not significant for the task at hand and in some cases it is difficult to know the exact place where the signature end and the disclaimer starts and vice versa. In emails like the example in Table 3.3 by ignoring the SIGNATURE segment we avoid a significant number of false positives.

Table 3.3: Example of email segmentation annotation

REPLY	Markowitz@DOCKMASTER.NCSC.MIL writes:
REPLY	
REPLY	> It is interesting to note in this regard that permission to export
REPLY	> PKZIP’s encryption scheme has twice been denied by NSA. Draw you own
REPLY	> conclusions.
REPLY	
OTHER	Uh, I’m afraid that your information is slightly out of date... PKWare
OTHER	has obtained a license to export their program to the whole world,
OTHER	except a very limited list of countries... Draw your own conclusions
OTHER	about the strength of the algorithm... :-)
OTHER	
SIGNATURE	Regards,
SIGNATURE	Vesselin
SIGNATURE	–
SIGNATURE	Vesselin Vladimirov Bontchev Virus Test Center, University of Hamburg
SIGNATURE	Tel.:+49-40-54715-224, Fax: +49-40-54715-226 Fachbereich Informatik - AGN
SIGNATURE	< PGP 2.2 public key available on request. > Vogt-Koelln-Strasse 30, rm. 107 C
SIGNATURE	e-mail: bontchev@fbihh.informatik.uni-hamburg.de D-2000 Hamburg 54, Germany
SIGNATURE	
SIGNATURE	This email contains information which may be confidential.
SIGNATURE	It is for the exclusive use of the intended recipient(s). If you are not the
SIGNATURE	intended recipient(s) do not copy this communication, or disclose it to any
SIGNATURE	other person. If you have received this email in error please notify the sender
SIGNATURE	immediately, delete the message from your computer system and destroy any copies.
SIGNATURE	
SIGNATURE	Except where this email is sent in the usual course of our business, any views
SIGNATURE	or opinions presented are solely those of the author and do not necessarily
SIGNATURE	represent those of University of Hamburg. Although University of Hamburg operates
SIGNATURE	anti-virus programs, it does not accept responsibility for any damage whatsoever
SIGNATURE	that is caused by viruses being passed. Replies to this email may be monitored
SIGNATURE	and/or recorded by University of Hamburg for operational or business reasons.

The previous messages are complete previous messages or just parts of it. They are frequently incomplete sentences that are interrupted by a response of the current email. Unlike the signature and the disclaimers, the Reply section should not be ignored, just treated differently.

Table 3.4: Email segments lines count

	Other	**Reply**	**Signature**	**Total**
Count	172347	9600	3707	185654

We constructed a dataset for email segmentation. We are using 2979 emails from various sources and different years to help the model generalize. The dataset contains 172347 lines marked as “OTHER”, 9600 marked as “REPLY” and 3707 marked as “SIGNATURE”, as shown in Table 3.4. One person annotated the emails for the email segmentation dataset for four months. The content was normalized using the ICU4J library so that we would have a more limited set of Unicode.

3.4.3 Data format

We have followed the same strategy for both datasets, email segmentation and PII detection, we have annotated. The email corpus was divided per email, and each email was saved in JSONL format to be fed to Prodigy to be annotated. In the case of PII annotation, we had gone one step further and divided each email into sentences, and those corresponded to JSON lines in the file used by Prodigy. The output of Prodigy after annotation was a similar format as the one used by spaCy, in JSON format, with the tokenization already done, as shown in the Example 3.1. The JSON format is straightforward to use as there is no shortage of good quality libraries to process JSON.

$$\begin{gathered}
\{\ \text{“text”: “My”, “start”: 660, “end”: 662, “id”: 130}\ \}, \\
\{\ \text{“text”: “suggestion”, “start”: 663, “end”: 673, “id”: 131}\ \}, \\
\{\ \text{“text”: “for”, “start”: 674, “end”: 677, “id”: 132}\ \}, \\
\{\ \text{“text”: “where”, “start”: 678, “end”: 683, “id”: 133}\ \}, \\
\{\ \text{“text”: “to”, “start”: 684, “end”: 686, “id”: 134}\ \}, \\
\{\ \text{“text”: “go”, “start”: 687, “end”: 689, “id”: 135}\ \}, \\
\{\ \text{“text”: “is”, “start”: 690, “end”: 692, “id”: 136}\ \}, \\
\{\ \text{“text”: “Austin”, “start”: 693, “end”: 699, “id”: 137}\ \},
\end{gathered} \tag{3.1}$$

The annotations are on a separate field of the record, as shown in Example 3.2. Each annotation include an id reference to the token where it starts and the token where it ends. And the information of the subset location in the original string.

{ “label”: “GPE”, “start”: 693, “end”: 699, “token_start”: 137, “token_end”: 137 }, (3.2)

3.4.4 Tag Representation

For the PII annotation, the data output from Prodigy was in BILUO notation. The BILOU notation specifies that single token named entities are labelled with U-XXXX where XXXX is the name of the named entity, such as PER for Person or LOC for locations. Multi-token named entities are marked with B-XXXX for the first token, I-XXXX for any inner tokens (if present) and L-XXXX for the last tokens. All other tokens that are not named entities are labelled with O.

Another labelling scheme we used was the BIOES (Begin, Inside, Outside, End, Single) that is similar to the BILOU scheme, as shown in Table 3.5, but some libraries only support one of the two notations.

There are other notations, such as the BIO (Begin, In, Out) notation, but previous works show that the models trained on those labels perform worse [67].

Table 3.5: Tag representations

BIOES	**BILUO**
Cordially O	Cordially O
, O	, O
Susan B-PERSON	Susan B-PERSON
S. I-PERSON	S. I-PERSON
Bailey E-PERSON	Bailey L-PERSON
Director S-TITLE	Director U-TITLE
Enron B-ORG	Enron B-ORG
North I-ORG	North I-ORG
America I-ORG	America I-ORG
Corp. E-ORG	Corp. L-ORG
1400 B-ADDRESS	1400 B-ADDRESS
Smith I-ADDRESS	Smith I-ADDRESS
Street I-ADDRESS	Street I-ADDRESS
, I-ADDRESS	, I-ADDRESS
Suite I-ADDRESS	Suite I-ADDRESS
3803A I-ADDRESS	3803A I-ADDRESS
Houston I-ADDRESS	Houston I-ADDRESS
, I-ADDRESS	, I-ADDRESS
Texas I-ADDRESS	Texas I-ADDRESS
77002 E-ADDRESS	77002 L-ADDRESS
Phone O	Phone O
: O	: O
(B-PHONE_NUMBER	(B-PHONE_NUMBER
713 I-PHONE_NUMBER	713 I-PHONE_NUMBER
) I-PHONE_NUMBER	) I-PHONE_NUMBER
853 I-PHONE_NUMBER	853 I-PHONE_NUMBER
- I-PHONE_NUMBER	- I-PHONE_NUMBER
4737 E-PHONE_NUMBER	4737 L-PHONE_NUMBER

Chapter 4

PII detection system

In this chapter, we describe the models that were developed and evaluated for email segmentation and for PII detection. First, we describe the models used for email segmentation and we analyse how the different models compare with each other, both in terms of performance and machine resources allocated. Then we describe and analyse the models we developed for PII detection and each model strengths and weaknesses.

4.1 Email Segmentation

For email segmentation, we constructed five different models (Table 4.1) exploring the balance between model size and accuracy. All five models are Recurrent Neural Networks, three of them were Elman Recurrent Neural Networks, one is an LSTM, and the last one is a Bi-LSTM with attention. As input to the email segmentation we used character embeddings trained on the dataset.

Table 4.1: Email Segmentation models

Model Name	**Vocab size**	**Model Size (MB)**	**Train acc.**	**Val. acc.**	**CPS***
Elman RNN	4232	1.8	66.28	59.18	225
Elman RNN	6346	2.6	59.78	56.78	229
Elman RNN	50 431	20.2	90.18	88.97	234
LSTM	50 431	20.5	93.50	94.63	33.83
BiLSTM	50 431	47.7	**97.28**	**94.75**	12.9

*chars per second

For the line segmentation models, we used character embeddings as input, that were concatenated to represent a line, as shown in Figure 4.1. Character embeddings were used instead of word-based ones because the characters at the beginning of the line are more significant than words for classifying lines as email segments.

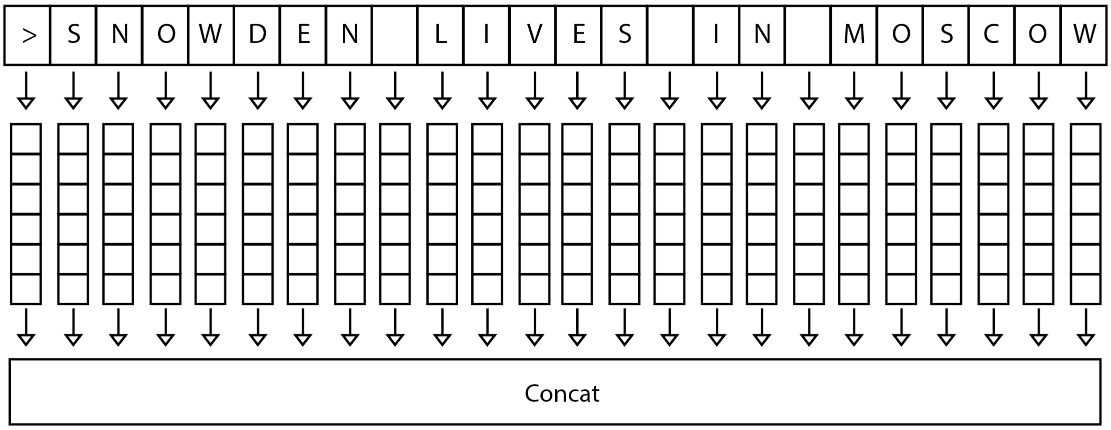


Figure 4.1: Line encoding for email segmentation models

As we train our models to perform better accordingly to a specific metric, the model might focus on features that are too specific to our training data. In that situation, a model overfits, leading to poor performance on unseen data. A common practice to reduce the risk of the model overfitting is to use regularisation. There are many regularisation strategies, and we used dropout [34, 72], a technique that consists of randomly dropping some neurons (setting them to 0) and their connections in a specific layer. Dropout was essential during training to help the model not overfit. However, dropout is only relevant during training. After the model is ready, we disable it during prediction time, or it would not be deterministic.

As the activation function, we used a Rectified Linear Unit (ReLU), as it produces good results in combination with dropout [17].

We used CrossEntropyLoss loss function using weights for different classes, since the email parts don't have the same distribution.

We used Adam [42] as the optimiser to select the learning rate for each minibatch, in combination with ReduceLROnPlateau scheduler that reduces the learning rate when a metric has stopped improving. Models often benefit from decreasing the learning rate when learning stagnates.

In the model comparison for email segmentation shown in Table 4.1, there are a few things we must note. First, the model that presents the higher accuracy is the BiLSTM with 94.75. But that comes with a cost: the model size is bigger, but more importantly, the model is the slowest with 12.9 characters being processed per second. When we consider the requirements in Chapter 3 Table 3.1, we realise that that makes the model unusable as it would not respond to 95% of requests under five seconds.

The comparison between LSTM (Figure 4.2) and BiLSTM (Figure 4.3) models is interesting because a slight improvement in accuracy comes at the cost of a decrease of speed to only 38% of the LSTM model. The BiLSTM model differs from the LSTM by adding a second LSTM layer and an attention layer. The BiLSTM also increases the model size to more than the double than the LSTM model.

The processing speed of the Elman RNN (Figure 4.4) is more inline with what we would

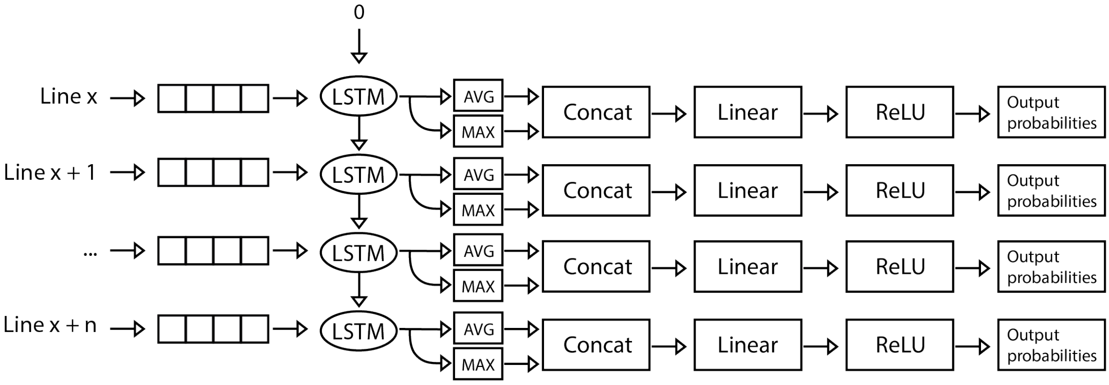


Figure 4.2: LSTM model for email segmentation

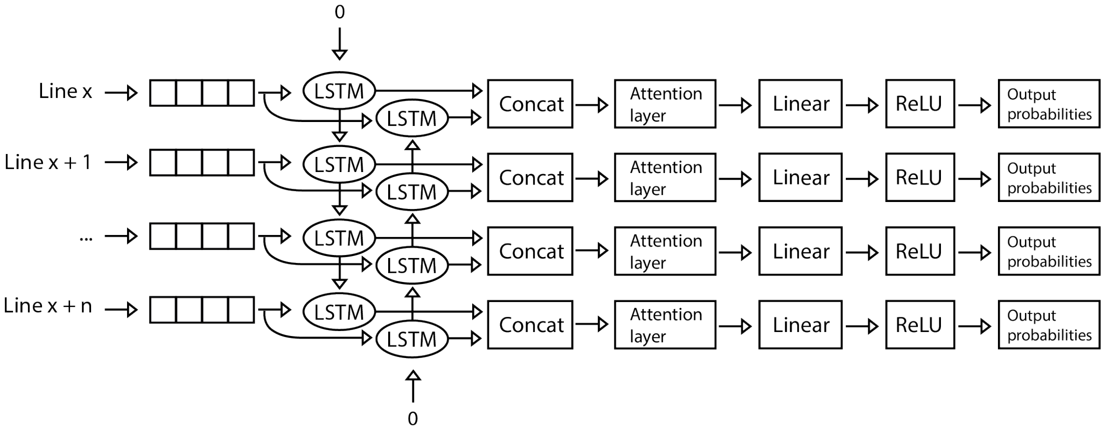


Figure 4.3: BiLSTM model for email segmentation

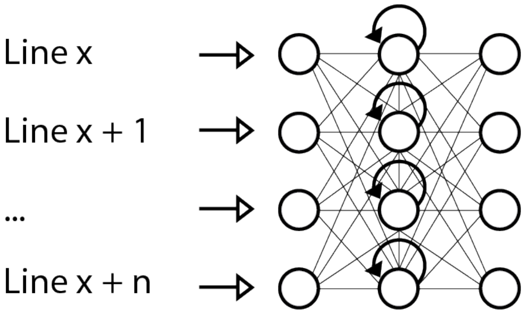


Figure 4.4: Elman RNN for email segmentation

expect to have in a production environment. Future research is needed to understand if either we can fix the bottlenecks in the BiLSTM model or if Elman RNN's accuracy can be improved, potentially with more data. The Elman RNN with a vocabulary of 50431 presents the best balance between the accuracy and the performance.

4.2 PII Detection

For PII detection in emails we have trained nine different models (Table 4.2), representing variations on two main architectures: a BiLSTM and a CNN based on the state-of-the-art for NER, as seen in Table 2.3. The number of named entities we have consider in these models is just a sub-set of the 30 named entities we have listed in Table 2.1. As mentioned in Table 3.2 we did focus on: Address, Email Address, Geo-Political Entity (GPE), IP Address, Organisation, Password, Person, Phone Number, Timex, Title and Username.

Table 4.2: PII detection models

#	***Model Name***	**Vocab size**[†]	**Model Size**[†]	**CPS**[*]	**Precision**	**Recall**	**Macro F1-Score**
1	CNN	6.2	4.0	132 303	86.557	85.746	86.150
2	CNN + GloVe vectors	985.7	4.2	137 301	87.264	86.909	87.086
3	CNN + GloVe vectors + orthography variance	985.8	4.2	147 517	86.867	86.769	86.818
4	CNN + GloVe vectors + drugs dataset	985.7	4.2	146 209	86.625	87.267	86.944
5	CNN + Bert uncased + drugs dataset	1376.2	4.0	147 992	86.575	85.547	86.058
6	CNN + Roberta base lg + drugs dataset	1331.2	4.0	134 717	86.575	85.547	86.058
7	BiLSTM + BPE	27.3	3.1	33 934	84.690	86.040	81.721
8	BiLSTM + Glove vectors + Flair embeddings	1863.7	368.6	141	87.550	90.040	**87.209**
9	BiLSTM + BPE + Glove vectors + Flair embeddings	2641.9	133.1	122	87.560	89.860	85.810

[*] chars per second
[†] in MB

The BiLSTM model 4.5 was used in the models # 7, 8 and 9 of the Table 4.2. In models # 8 and 9 we used Glove [61] and Flair embeddings [4] .

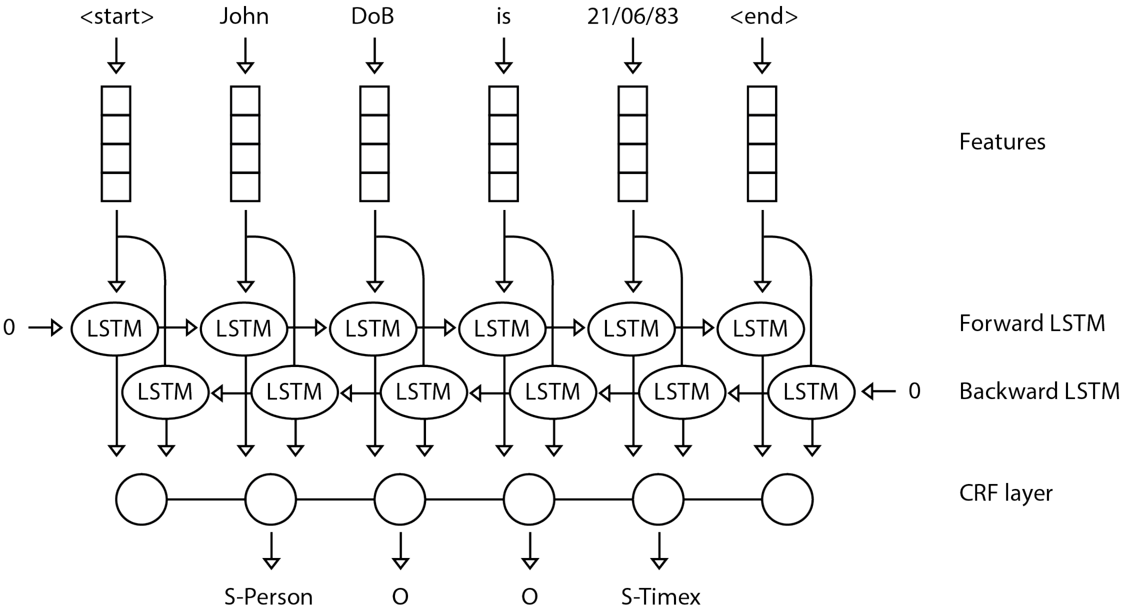


Figure 4.5: BiLSTM model for PII detection

The model size and speed are essential metrics as defined in the requirements in Table 3.1. However, in a similar way to what we have seen for the email segmentation model, the BiLSTM model precision and recall comes at the cost of the model size and processing speed. We compare the model with the other architecture also in state-of-the-art, a Convolution Neural Network (CNN).

The first CNN model (Figure 4.6) does not use any pre-trained embeddings, just the embeddings trained on the dataset. The CNN achieves a macro f1-score of 86.150, close to the value obtained by the BiLSTM of 87.209. While the CNN has a lower precision and recall than the BiLSTM model, the speed is 938 times higher on the same CPU. The CNN model also has lower memory requirements for both the model and the vocabulary used. We trained a second model adding Glove vectors to the simple CNN model: the macro f1-score increases from 86.150 to 87.086, closer to the value achieved by the BiLSTM. The speed of the model also increases slightly compared to the first CNN, continuing to be much higher than the BiLSTM.

Analysing the discrepancy in metrics between the training and the evaluation datasets (in Table 4.2 we just present the values from the validation set), we suspect that using a data augmentation technique might help to improve the model score. We apply a data augmentation technique, by automatically introducing a small number of typos in the dataset during training, following previous work by Bojanowski et al. [11]. The resulting model, a CNN with glove vectors and using orthography variance, did not have improved metrics, either precision, recall or macro f1-score, lowering the macro f1-score to 86.818.

As mentioned before, the number of named entities we have consider in these models is just

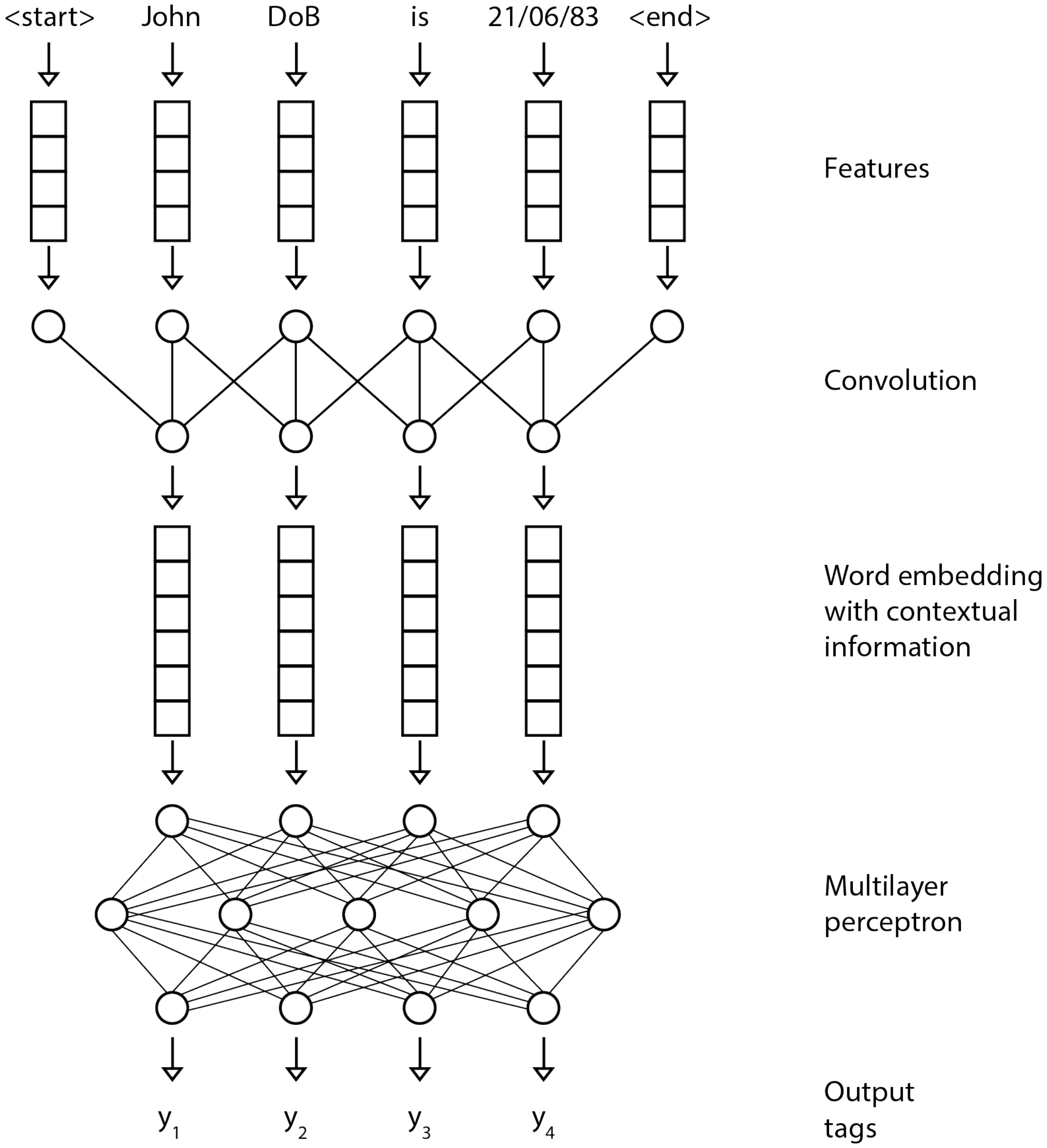


Figure 4.6: CNN model for PII detection

a sub-set of the 30 named entities we have listed in Table 2.1. The question is if the metrics we are evaluating are affected if we would cover more types of PII. To evaluate that, we have added a set of emails with annotated drug names. We trained a version of the previous CNN model with Glove vectors including this time the new drugs data. The results are slightly worse: the F1 score decreases from 87.086 to 86.944. The impact was lower than the usage of orthography variance data augmentation technique.

We have experimented with BERT embeddings and RoBERTa embeddings on the same CNN architecture but they did not improve the results when compared to the CNN + Glove embeddings model.

Going back to the BiLSTM model, we evaluated what would be the impact of using the Byte-Pair-Encoding (BPE) subword embedding in the metrics of the BiLSTM model. The speed of the model in characters per second was the best we achieved with the BiLSTM architecture, but still several times slower than any CNN model we trained. The macro F1 score was just 81.721, the lower F1 score in this comparison.

Finally we tested if the BPE subword embeddings improved the existing BiLSTM model that was using Glove and Flair embeddings. The model did improve precision from 87.550 to 87.550 but decreased the recall to 89.860, and a lower macro F1 score of 85.810. This model was the slowest model in this comparison, only processing 122 characters per second during inference.

4.2.1 Regular Expressions

During the various models that we trained, the TIMEX PII type was consistently one of the elements with worst prediction. Being a named entity that is mainly represented by fixed formats and numeric representations, the question was if we would be better off by detecting those using regular expressions.

We did evaluate two different versions of TIMEX detection from the previous system that was already used in the company, trying to balance precision and recall. They did perform significantly worse than our BiLSTM model, as shown in Table 4.3, and it was a laborious task of adapting the regex to a substantial number of cases that were present in the training dataset.

Table 4.3: Comparison between machine learning model and regular expressions

	Precision	**Recall**	**F1-Score**
Regular Expression v1	89.073	69.899	78.329
Regular Expression v2	94.941	59.610	73.237
BiLSTM + Glove vectors + Flair embeddings	83.970	87.100	**85.510**

The regular expressions are a easy way to get quick results but as the number of different cases and the complexity increases it becomes harder to manage manually. The machine learning models simply the process by automatically balancing the model to get a better F1-score across the examples.

4.3 Discussion

The usage of the state-of-art BiLSTM architecture using pre-trained embeddings comes at a cost of speed of memory usage that makes the model not adequate for a production environment. The difference between the metrics of the best BiLSTM model and the best CNN model that we trained suggest that we would be better off working in improving the CNN model than trying to make the BiLSTM more performant.

Chapter 5

Conclusions and Future Work

PII detection in email content is an increasingly relevant topic with practical applications in information technology system, as a response to increasing regulations. The global pandemic we are facing in 2020 is increasing the digitisation of companies. With more people working from home, remote collaborations increased the already significant amount of information process via email. But companies also faced an increased amount of attacks [56, 30]. For all these factors, PII detection and protection will continue to be a relevant research topic. The aim of PII detection in email is twofold. First, to prevent PII information from leaving the company, or from being sent to someone inside the company that should not have access to it. Secondly, to serve as a tool to find PII information present in email archives. Both of these tasks are requirements imposed by the GDPR to every company that operates in the European common market. The detection of PII is the basis of many other tasks, such as anonymisation, archive search and e-discovery. Typical approaches to PII detection, such as the one implemented in Microsoft Office 365, involve the usage of regex and gazetteers, offering a limited coverage of PII types that are defined in the GDPR. In this dissertation we address two issues of PII detection in emails, namely how to reduce the number of false positives in email signatures and replies, and what machine learning architecture is more suitable for running in a production environment for PII detection.

5.1 Challenges

In this section we describe some of the challenges that we had to tackle throughout the development of this dissertation. Without available dataset of annotated PII in emails we had to annotate a significant number of emails for the 12 PII types we used. Different types of PII present diverse challenges, and maintaining consistency in the annotation is hard. As we went through the annotation process we learned more about the annotation and wanted to adjust our strategy. A PII dataset is very unbalanced and it is hard to find examples for some of the types of data. While the

PII named entities present their own challenges, the email medium presents an additional problem. The free text structure, where both casual and corporate language is used, the lack of formal structure of the text, the discrepancy in text sizes and the common misspellings are all challenges that were avoided in most systems available in the state-of-art for Named Entity Recognition. By realising that the state-of-art Named Entity Recognition systems were not adequate, based on the established metrics, the search for an architecture that would supply a better balance between the metrics was cumbersome. Additional problems were found in trying to adjust the data to the different tools used, with their own input/output types.

5.2 Further Work

In future work we expect an improvement in the email segmentation model that would tackle also the HTML content of emails. The balance in the email segmentation model can probably be improved with some manual features, and that can be investigated in future works. The PII detection model could get improved with additional PII types and more annotated data. The vast number of possible PII entities inside the same model will present a maintenance challenge. How to maintain the model and continuously improve it in an efficient way will be an interesting topic. The two models operate separately, with the email segmentation model just filtering the disclaimer and signature parts from the email. With that the number of false positives of PII detection model decreases. Future work could look into integrating the two models into the same model, decreasing the processing time and potentially improving the other metrics.

Co-reference resolution could be used to establish the relationship between the PII named-entities found in the text. The co-reference is essential in the detection of contextual PII (Table 2.1), as some named-entities are only PII when they connected with other pieces of information that allow to specifically identify a person. The previous messages in email presents an additional challenge, because they are often cut in parts, so investigation would be required to understand how to establish a reference connection between an element in a complete sentence and a partial sentence. The email segmentation model can be used to detect the previous messages parts and based on that the logic for doing the co-reference can be different than the one used for the main text.

Appendix A

Tokenisation Libraries

Table A.1: Tokenisers comparison

Tokeniser	**Tokenization Type**	**Programming Languages**	**Languages**	**Used by**
Europarl	Sentence level Word level	Perl	German Greek English Spanish French Italian Portuguese Swedish	FastText
HuggingFace	Byte level BPE* Char level BPE* Subword level Word level	Rust Python Node.js		AllenNLP
ICU	Character level Word level	C++ Java	Chinese Hebrew Japanese Khmer Korean Lao Latin Myanmar	FastText
KyTea	Word level	C++	Chinese Japanese	Sentencepiece
Mecab	Word level	Perl Python Ruby Java	Chinese Japanese	Sentencepiece FastText
Sentencepiece	BPE* Subword level Char level Word level	C++ Python	Language Independent	Flair
Spacy	Word level Sentence level	Python	55 languages including Portuguese	AllenNLP Flair OpenAI GPT
Stanford Tokeniser	Word level	Java	English French Spanish	
UETsegmenter	Word level	Java	Vietnamese	FastText
Wordpiece	BPE*	Python	Language independent	Bert and derivatives

* byte-pair-encoding (BPE)

Appendix B

NER Corpora genres

Table B.1: NER Corpora genres and number of NEs of different class

CORPUS	NE #	YEAR	LANGUAGES	NAMED ENTITIES TYPES
MUC-6 [75]	1.129	1995	English	date, location, monetary amount, organisation, percentage, person, time
MUC-7	1.987	1996	English	date, location, monetary amount, organisation, percentage, person, time
MET-2	7.113	1996	Chinese, Japanese	date, location, monetary amount, organisation, percentage, person, time
NIST IEER	5.002	1999	English, Mandarin	cardinal, date, duration, location, measure, money amount, organisation, percentage, person, time
Egunkaria [3]	4.748	2000	Basque	location, miscellaneous, organisation, person
CoNLL 2002	46.607	2002	Dutch, Spanish	location, miscellaneous, organisation, person
CoNLL 2003[78]	71.045	2003	English, German	location, miscellaneous, organisation, person
ACE 2004	6.903 [33]	2004	English, Chinese, Arabic	facility, geographical/political, location, organisation, person, vehicle, weapon
HAREM [70]	7.817	2004	Portuguese	abstraction, event, location, measure, miscellaneous, organisation, person, product, thing, time
BBN [66]	99.265 [1]	2005	English	contact-info, event, facility, gpe, language, law, location, nationality, organisation, person, product, work of art
HAREM 2 [28]	$\sim$ 40.000	2010	Portuguese	abstraction, event, location, measure, miscellaneous, organisation, person, product, thing, time
WikiNer [58]	305.460 [69]	2012	Dutch, English, French, German, Italian, Polish, Portuguese, Russian, Spanish	location, miscellaneous, organisation, person.
GermEval [9]	41.005	2014	German	location, organisation, person, other

References

[1] Abhishek Abhishek, Ashish Anand, and Amit Awekar. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807, Valencia, Spain, April 2017. Association for Computational Linguistics.

[2] Rodrigo Agerri and German Rigau. Robust multilingual named entity recognition with shallow semi-supervised features. *Artificial Intelligence*, 238:63 – 82, 2016.

[3] Eneko Agirre, Elena Garcia, Mikel Lersundi, David Martinez, and Eli Pociello. The basque task: Did systems perform in the upperbound? In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 9–12, 2001.

[4] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018: 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[5] Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. Cloze-driven pretraining of self-attention networks. In *2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

[6] Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. Overview of the trec 2010 entity track. In *NIST Special Publication*, 2010.

[7] Valentin Barriere and Amaury Fouret. May I check again? — a simple but efficient way to generate and use contextual dictionaries for named entity recognition. application to French legal texts. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 327–332, Turku, Finland, September–October 2019. Linköping University Electronic Press.

[8] Patrice Bellot, Véronique Moriceau, Josiane Mothe, Eric SanJuan, and Xavier Tannier. Inex tweet contextualization task: Evaluation, results and lesson learned. *Information Processing & Management*, 52(5):801 – 819, 2016.

[9] Darina Benikova, Chris Biemann, and Marc Reznicek. Nosta-d named entity annotation for german: Guidelines and dataset. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA).

[10] Jules Berman. Comparing de-identification methods. https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/1472-6947-6-12/comments, 2006. [Online; accessed September 2019].

[11] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[12] Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *CoRR*, abs/1511.08308, 2015.

[13] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[14] Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[15] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(76):2493–2537, 2011.

[16] Savelie Cornegruta, Robert Bakewell, Samuel Withey, and Giovanni Montana. Modelling radiological language with bidirectional long short-term memory networks. In *Proceedings of the Seventh International Workshop on Health Text Mining and Information Analysis*, pages 17–27, 01 2016.

[17] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8609–8613, 2013.

[18] Vitor Rocha de Carvalho and William W. Cohen. Learning to extract signature and reply lines from email. In *CEAS*, 2004.

[19] Gianluca Demartini, Tereza Iofciu, and Arjen P. De Vries. Overview of the inex 2009 entity ranking track. *INEX'09 Proceedings of the Focused retrieval and evaluation, and 8th international conference on Initiative for the evaluation of XML retrieval*, 6203:254–264, 2009.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, 2019.

[21] Francesco Di Cerbo and Slim Trabelsi. Towards personal data identification and anonymization using machine learning techniques. In András Benczúr, Bernhard Thalheim, Tomáš Horváth, Silvia Chiusano, Tania Cerquitelli, Csaba Sidló, and Peter Z. Revesz, editors, *New Trends in Databases and Information Systems*, pages 118–126, Cham, 2018. Springer International Publishing.

[22] George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. The automatic content extraction (ace) program tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-2004)*, 2004.

[23] Jacob Eisenstein. *Introduction to Natural Language Processing*. MIT press, 2019.

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

[25] EU Science Hub. JRC-Names. https://ec.europa.eu/jrc/en/language-technologies/jrc-names, 2019. [Online; accessed 11-September-2019].

[26] International Organization for Standardization. Iso 27001 - information technology — security techniques — information security management systems — requirements. https://www.iso.org/isoiec-27001-information-security.html, 2013. [Online; accessed 11-September-2019].

[27] International Organization for Standardization. Iso 8601 date and time format. https://www.iso.org/iso-8601-date-and-time-format.html, 2019. [Online; accessed 11-September-2019].

[28] Cláudia Freitas, Cristina Mota, Diana Santos, Hugo Gonçalo Oliveira, and Paula Carvalho. Second harem: Advancing the state of the art of named entity recognition in portuguese. In *LREC*, 2010.

[29] Abbas Ghaddar and Philippe Langlais. Robust lexical features for improved neural network named-entity recognition. *CoRR*, abs/1806.03489, 2018.

[30] Randi Gollin. Cyber threats are surging as employees work from home due to the covid-19 pandemic, prompting cybersecurity insurers to reassess companies' security measures—and potentially raise premiums. https://www.mimecast.com/blog/2020/06/the-impact-of-covid-19-on-cyber-security-insurance/. Accessed: 2020-06-08.

[31] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 4, pages 2047–2052 vol. 4, July 2005.

[32] Ralph Grishman and Beth Sundheim. Message understanding conference-6: a brief history. In *COLING '96 Proceedings of the 16th conference on Computational linguistics - Volume 1*, volume 1, pages 466–471, 1996.

[33] B. HACHEY, C. GROVER, and R. TOBIN. Datasets for generic relation extraction. *Natural Language Engineering*, 18(1):21–59, 2012.

[34] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv e-prints*, page arXiv:1207.0580, July 2012.

[35] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. Master's thesis, TU Munich, 1991. Diploma Thesis.

[36] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[37] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015.

[38] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

[39] Allen Kent, Madeline M. Berry, Fred U. Luehrs Jr., and J. W. Perry. Machine literature searching viii. operational criteria for designing information retrieval systems. *American Documentation*, 6(2):93–101, 1955.

[40] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 284–294, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[41] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2741–2749. AAAI Press, 2016.

[42] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[43] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *Proceedings of the 15th European Conference on Machine Learning*, ECML'04, page 217–226, Berlin, Heidelberg, 2004. Springer-Verlag.

[44] Cvetana Krstev, Ivan Obradović, Miloš Utvić, and Duško Vitas. A system for named entity recognition based on local grammars. *Journal of Logic and Computation*, 24(2):473–489, 02 2013.

[45] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001.

[46] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *CoRR*, abs/1603.01360, 2016.

[47] Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramón Fermandez, Silvio Amir, Luís Marujo, and Tiago Luís. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[48] Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. Empower sequence labeling with task-aware neural language model. *CoRR*, abs/1709.04109, 2017.

[49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[50] Xuezhe Ma and Eduard H. Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *CoRR*, abs/1603.01354, 2016.

[51] Marshall McLuhan. *Understanding Media: The Extensions of Man*. McGraw-Hill Book Company, 1964.

[52] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[53] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[54] Einat Minkov, Richard C. Wang, and William W. Cohen. Extracting personal names from email: Applying named entity recognition to informal text. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 443–450, 2005.

[55] Ishna Neamatullah, Margaret M Douglass, Li wei H Lehman, Andrew Tomas Reisner, Mauricio Villarroel, William J Long, Peter Szolovits, George B Moody, Roger Greenwood Mark, and Gari D Clifford. Automated de-identification of free-text medical records. *BMC Medical Informatics and Decision Making*, 8(1):32–32, 2008.

[56] Nexusguard. Ddos threat report 2020 q1. https://blog.nexusguard.com/threat-report/ddos-threat-report-2020-q1. Accessed: 2020-06-10.

[57] Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian (Allen) Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. Semantic product search. In *KDD '19 Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2876–2885, 2019.

[58] Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151 – 175, 2013. Artificial Intelligence, Wikipedia and Semi-Structured Resources.

[59] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707, 2016.

[60] The European Parliament and the Council of the European Union. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation), 2016.

[61] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[62] Georgios Petasis, Alessandro Cucchiarelli, Paola Velardi, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D. Spyropoulos. Automatic adaptation of proper noun dictionaries through cooperation of machine learning and probabilistic methods. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 128–135, 2000.

[63] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[64] Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *CoRR*, abs/1705.00108, 2017.

[65] Aleksandra Piktus, Necati Bora Edizel, Piotr Bojanowski, Edouard Grave, Rui Ferreira, and Fabrizio Silvestri. Misspelling oblivious word embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3226–3234, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[66] Ralph Weischedel, Ada Brunstein. BBN Pronoun Coreference and Entity Type Corpus. `https://catalog.ldc.upenn.edu/LDC2005T33`, 2005. [Online; accessed 09-September-2019].

[67] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[68] Gil Rocha, Christian Stab, Henrique Lopes Cardoso, and Iryna Gurevych. Cross-lingual argumentative relation identification: from english to portuguese. In *Proceedings of the 5th Workshop on Argument Mining*, pages 144–154, 2018.

[69] Marc-Antoine Rondeau and Yi Su. Lstm-based neurocrfs for named entity recognition. In *Interspeech 2016*, pages 665–669, 2016.

[70] Diana Santos, Nuno Seco, Nuno Cardoso, and Rui Vilela. Harem: An advanced ner evaluation contest for portuguese. In *LREC*, pages 1986–1991, 2006.

[71] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1715–1725, 2016.

[72] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.

[73] Jana Straková, Milan Straka, and Jan Hajic. Neural architectures for nested NER through linearization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy, July 2019. Association for Computational Linguistics.

[74] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. ERNIE 2.0: A continual pre-training framework for language understanding. *CoRR*, abs/1907.12412, 2019.

[75] Beth M. Sundheim. Overview of results of the muc-6 evaluation. In *TIPSTER TEXT PROGRAM PHASE II: Proceedings of a Workshop held at Vienna, Virginia, May 6-8, 1996*, pages 423–442, Vienna, Virginia, USA, May 1996. Association for Computational Linguistics.

[76] Charles Sutton. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

[77] Wilson L. Taylor. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, 1953.

[78] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*. Edmonton, Canada, 2003.

[79] Peter D. Turney and Patrick Pantel. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010.

[80] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS'17 Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.

[81] Genta Indra Winata, Zhaojiang Lin, Jamin Shin, Zihan Liu, and Pascale Fung. Hierarchical meta-embeddings for code-switching named entity recognition. In *2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

[82] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

[83] Minghao Wu, Fei Liu, and Trevor Cohn. Evaluating the utility of hand-crafted features in sequence labelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2850–2856, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[84] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv e-prints*, page arXiv:1609.08144, September 2016.

[85] Jie Yang and Yue Zhang. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations*, pages 74–79, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[86] Jie Yang, Yue Zhang, and Fei Dong. Neural reranking for named entity recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 784–792, Varna, Bulgaria, September 2017. INCOMA Ltd.

[87] Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks. *CoRR*, abs/1703.06345, 2017.

[88] Zhixiu Ye and Zhen-Hua Ling. Hybrid semi-Markov CRF for neural sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–240, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[89] Wenpeng Yin and Hinrich Schütze. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[90] Zi Yin and Yuanyuan Shen. On the dimensionality of word embedding. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 895–906, Red Hook, NY, USA, 2018. Curran Associates Inc.

[91] Shanshan Zhang, Lihong He, Eduard C. Dragut, and Slobodan Vucetic. How to invest my time: Lessons from human-in-the-loop entity extraction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '19*, pages 2305–2313, 2019.

[92] Qile Zhu, Xiaolin Li, Ana Conesa, and Cécile Pereira. GRAM-CNN: a deep learning approach with local context for named entity recognition in biomedical text. *Bioinformatics*, 34(9):1547–1554, 12 2017.