Business Analytics Master Thesis

De-identifying Personal Information in Dutch Mortgage Documents using Named Entity Recognition

by

Tim Vorstenbosch
(tvh900 (2588989))

Home Organisation :	Vrije Universiteit	*Host Organisation :*	Stater N.V.
Departement :	Business Analytics	*Department :*	Datalab
Street :	De Boelelaan 1111	*Street :*	Podium 1
Postcode :	1081 HV	*Postcode :*	3826 PA
City	Amsterdam	*City :*	Amersfoort
First supervisor :	Sandjai Bhulai	*Daily supervisor :*	Roy Rops
Second reader :	Rob van der Mei		

August 29, 2022

A thesis submitted in fulfilment of the requirements for
the Computational Intelligence track of the Master Business Analytics degree

"What is it that makes us human? The answer that came to me again and again was play. Every human society in recorded history has games. We don't just solve problems out of necessity, we do it for fun, even as adults. Leave a human being alone with a knotted rope and they will unravel it. Leave a human being alone with blocks and they will build something. Games are part of what makes us human. We see the world as a mystery, a puzzle, because we've always been a species of problem-solvers"
The Talos Principle: Alexandra Drennan's time capsules by Erin Fitzgerald

Preface

This master thesis will discuss the technique of Named Entity Recognition on unstructured text using deep learning models. In other words, this is an adaptation of multiple courses that were given in the BA program. The most crucial and relevant courses related to this master thesis are Advanced Machine Learning, Deep Learning, Natural Language Processing Technology and Applied Text Mining. The purpose of this thesis is to investigate this technique and its recent Deep Learning adaptations.

This thesis will investigate to what extent it is possible to use the technique of NER for the de-identification of the financial documents given the requirements by both Stater and the GDPR regulation. Both the tasks of anonymisation and pseudonymisation are part of the de-identification task and hence both will be considered. This research was conducted at the company Stater, which is the largest mortgage service company in the Netherlands. The department in which this research was carried out is the Datalab, a division of the Data Services team specifically aimed to solve data science problems such as these.

Acknowledgements

I am profoundly grateful to everyone from the Vrije Universiteit who has provided me the opportunity to carry out my research. The person that I want to thank first and foremost from this organisation is my first supervisor Sandjai Bhulai, who has always cheerfully aided and supported me with this thesis throughout these past few months. I also want to thank him for his flexibility, honest feedback and the exciting brainstorming sections during this internship. Second, I want to thank Annemieke van Goor, the Business Analytics Internship Coordinator, for speedily providing me with answers to the various questions I had both before and during my master thesis. Lastly, I want to thank Rob van der Mei and Eduard Belitser for supporting me and taking time out of their day to help set-up this project.

I also want to express my gratitude to the host organisation Stater for allowing me to carry out this master thesis at their company. In particular, I want to thank Roy Rops, the Lead Datalab, for handling most of the organisational matters of my internship and for providing me with the opportunity to do this internship at Stater. I also want to thank Adri Geuze, and Jens Holland, both colleagues within the Datalab, for continually helping me with the data for this massive project and providing me with the necessary information and tools to understand en process this data. Furthermore, I want to thank Ruben Westerhof, the lead Data Services, for handling various miscellaneous organisational matter and allowing me to present my thesis to the other departments. Lastly, I want to thank the entire Datalab for their honest and useful feedback during my thesis and for the fun learning environment of my internship.

Lastly, I want to thank my parents, Loes and Louis, for continuously supporting me throughout these hectic times. I also want to thank Marije Gemmink for her continuous support and brainstorming sessions during the internship.

Executive Summary

This thesis project investigated the technique of Named Entity Recognition (NER) for the task of document de-identification within the domain of financial services. With the recent introduction of the General Data Protection Regulation (GDPR), the rules surrounding private and sensitive information have become more strict in recent years. As a result, it is crucial for Stater to properly adapt to such new regulations. Hence, the Datalab department can greatly benefit from a tool that is able to de-identify information in documents, since then many of these new regulations do not apply as strictly to these documents. Additionally, it will provide the Datalab department with extra security measures for their privacy-sensitive data, which is beneficial to the already great reputation of Stater. The technique of NER is able to find the most important words in a text and give them a label. Consequently, it will be interesting to investigate to what extent this technique can both anonymise and pseudonymise the documents processed by Stater.

In order to investigate this technique for the task of de-identification, it is important to define the terms anonymisation and pseudonymisation. Anonymisation refers to the removal of privacy-sensitive information, such that this information cannot be re-identified. Pseudonymisation refers to the task of replacing the privacy-sensitive data by other data that is synthetically created. The main difference between these two de-identification methods is that pseudonymisation allows the re-identification of privacy-sensitive information through protected keys and/or numbers. After defining these terms and searching the literature for potential NER models and techniques, the data was processed and annotated, so that the NER models can be trained. Two types of data were used, namely e-mail texts and inquiry documents. E-mail texts were mainly used to train the NER models on text, while the inquiry documents were used to train the NER models on table-based data. The specific data directories used for this project will be provided separately from this document for confidentiality reasons. For the annotation process, an annotation tool was manually created to both save costs for Stater and to vastly speed up this process. Stater is able to use this tool for other applications and can customise this tool further to their preferences. In the end, about 30,000 sentences were annotated for the models.

Three Deep Learning-based models were tested in this thesis, namely SpaCy, BERTje and RobBERT and experiments were performed with a total of 8 variations of these main models. These variations include the usage of a regular expression (Regex) system and a Conditional Random Field (CRF) model. In the end, all three models performed admirably in different scenarios, where the best models were able to nearly equal the best NER scores for the Dutch language within the literature (84.2% compared to $\approx$ 86% from the literature). The SpaCy model, in combination with the Regex system, performed the best for exact matches and labelling of words. In contrast, BERTje performed the best for anonymisation-based matches, which was evaluated using a manually created anonymisation score. RobBERT performed the best of the BERT-based models for this same score system if no further systems were introduced. Therefore, the best model depends on the use case of the NER models. Moreover, given their great generalisability, the scores are sufficiently high that they can be used in practice for the task of pseudonymisation. Despite these great scores, using only these models for the task of anonymisation is not sufficient. Hence, extra techniques, such as k-anonymisation and differential privacy, will have to be implemented in order to better de-identify these documents. Moreover, these scores can likely be further improved by having a larger amount of high quality and diverse data, so that the BERT-based models can be trained from scratch.

Table of Contents

List of Figures

List of Tables

De-identifying Personal Information in Dutch Mortgage Documents using Named Entity Recognition

Tim Vorstenbosch
Business Analytics
Vrije Universiteit Amsterdam
t.vorstenbosch@student.vu.nl
Student-ID: 2588989

Abstract

Named Entity Recognition (NER) has been used for many tasks, such as Question Answering, Sentiment Analysis and News Content Classification. However, one of the newer uses of this technique is for the anonymisation and pseudonymisation of documents. With the introduction of the GDPR guidelines in 2018, the popularity of both de-identification and NER techniques have increased rapidly for various languages and domains. The aim of this document is to help this research by using NER to de-identify the Dutch financial documents processed by Stater, which is the largest mortgage service company in the Netherlands. Hence, this thesis investigated to what extent the Dutch pre-trained NER models can be fine-tuned or new systems be created, such that these documents are de-identified and to what extent these documents conform to all requirements listed by both Stater and the GDPR regulation. To study this question, two types of data were manually annotated and three pre-trained models were tested, which are the Dutch SpaCy NER model, BERTje and RobBERT. Given the current data, annotation process and experiments, all three models show much potential in de-identifying the documents owned by Stater. These results and the challenges surrounding these results will be described in detail within this document.

1 Introduction

Data privacy is a sensitive yet crucial topic that has seen much development in the last decade due to the rise of big data [7] and the increased usage of web services [8]. As a result, various approaches have been proposed to help protect sensitive and personal information [8, 9, 10]. Due to the introduction of a new regulation named the General Data Protection Regulation (GDPR) [11], the research regarding data privacy has become even more prevalent, since much stricter security and privacy standards have been set for companies [12]. This data privacy regulation was put into effect on May 25 2018 [13] and its Dutch equivalent ("de Algemene Verordening Gegevensbescherming" (AVG)) was introduced on this same day. Although this new law is considered vital for data protection, it brought forth both positive effects, such as better data protection, and negative effects, such as a potential loss of important knowledge and increased security costs [14].

As a result, this new law has had a large effect on companies within the European Union, specifically those dealing with large amounts of sensitive and personal data. One of the largest sectors that deals with this type of data is the financial sector and one of the most prevalent businesses within the mortgage landscape is "Stater N.V.". Stater N.V. (or Stater for short) is currently the largest mortgage service company in the Netherlands and it manages 40% of all mortgages in this country [15]. Since Stater deals with large amounts of sensitive and personal data each day, this data cannot

be processed and distributed further without the client's (mortgage providers such as bank and insurance companies) and consumer's consent. Therefore, without this consent and given the current guidelines of the GDPR regulation, Therefore, the company is unable to use this data for analysis. Moreover, if data needs to be kept for a longer period of time, this is also not possible unless the data does not contain privacy sensitive information. Hence, a solution needs to be found for this problem.

The most prevalent method of dealing with documents that contain personal information is by using a technique called anonymisation, since this method falls outside the scope of GDPR [16]. This can be traced back to GDPR Recital 26, which states that these principles of data protection do not hold for data that has been properly anonymised [17]. Anonymised information is defined as follows in this EU recital: "information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable"[17]. Therefore, if the data conforms to this principle, Stater is more flexible with these rules and is able to use their information for other purposes.

Although the concept of anonymisation is relatively simple to understand, effectively performing this task is considered a challenge [16]. Additionally, given the rise of big data within companies, performing such tasks by hand has become impractical and learning from such data has become a major challenge in this field [18]. It would cost too much time, money and manpower to solve this problem completely by hand, given the enormous number of documents containing such sensitive information. For companies such as Stater, who deal with a variety of documents in various languages, this task has also become unmanageable. One of the few ways to make this task manageable for companies is to use software programs. Due to the advancements of technology in the past decades, such large amounts of data can be processed and analysed properly by enterprises with such programs [19]. A consequence of this advancement is the rise of various advanced analysis techniques such as machine learning [18].

One of the more upcoming techniques for solving the problem of anonymisation is Named Entity Recognition (NER). This technique is able to identify single or multi-word phrases and expressions in text, which includes persons, organisations and geo-political entities (GPE) [20, 21], which are referred to as named entities. These named entities are important to localise and classify, since these information units are crucial for various tasks, such as text understanding, automatic text summarisation, machine translation and anonymisation [22, 23, 20]. Despite its usefulness, little research has been performed in the domain of financial services on the topic of NER-based anonymisation techniques. Additionally, NER is typically applied to the English language and not the Dutch language [22]. Therefore, this thesis aims to investigate the topic of anonymising Dutch documents for the domain of financial services given its relevancy to companies such as Stater. The research question corresponding to this investigation is then as follows:

To what extent can existing Named Entity Recognition models be fine-tuned or new NER models and systems be created for locating, classifying and de-identifying private and delicate information inside the Dutch documents processed by Stater, such that the original content of these documents is kept intact while also conforming to the requirements listed in the GDPR?

This research question will be answered by answering each of the following sub-questions:

- What does it mean for a document to be de-identified, anonymised or pseudonymised?
- How well have Named Entity Recognition models performed in the past and what are their main shortcomings?
- How should the datasets provided by Stater be processed and annotated, such that the NER models can be properly trained and fine-tuned?
- To what extent can new or existing NER schema's adhere to the guidelines listed by both Stater and the GDPR regulation.
- Which pre-trained models and techniques have the most potential of successfully de-identifying the provided documents given the practical requirements provided by Stater?
- Can the models be used in practice given the current requirements of the de-identification definitions?
- How well do these models generalise to new entities and documents?

This thesis will start with a literature study in Section 2, where all of the definitions, related work and the task itself will be discussed in further detail. In Section 3, the documents provided by Stater and their preprocessing/selection steps are described. Section 4 will elaborate on the AVG/GDPR guidelines, the NER label schema, with which privacy-related information will be detected, and the results of the annotation process. Section 5 will describe the mathematics used in all of the models and gives information on how these models were trained, while Section 6 will specify the setup of the experiments performed on these models. Section 7 will provide the results of the experiment and Section 8 will investigate these results in more detail by analysing the errors made by the best models. The thesis will conclude with Sections 9 and 10, which will summarise the findings, discuss these findings and supply future researchers with possible improvements.

2 Literature Study

In this section, the literature study for this project will be reported. First, the origins and methodology of this literature study will be described. Second, the technique of Named Entity Recognition will be properly defined in both textual and mathematical terms. Third, the definitions for anonymisation, pseudonymisation and de-identification will be presented, their differences explained and their relation to NER is analysed. Next, the history and evolution of Named Entity Recognition as a task will be specified in terms of the embeddings, features and models used throughout the years. Last, the progress of these systems for the Dutch language will be investigated. Due to the novelty of this thesis in terms of domain, little to no literature is found on the task of de-identification within the financial-services domain. Therefore, most research is done on the task of named entity recognition and on the concept of de-identification. Consequently, it is assumed that the current literature of the NER task can be adapted for the financial domain, since there is a significant amount of overlap in terms of labels and techniques between the various NER domains. However, the degree of overlap will depend on the models and techniques chosen, since rule-based techniques, for instance, are more difficult to generalise in comparison to deep learning models. The methodology behind generalising these techniques to the financial domain will be described later in the Methodology section.

2.1 Meta Analysis

The most prevalent search engine for this project was Google Scholar, since such an engine is ideal for finding a wide variety of papers and books that are related to, for instance, the task of Named Entity Recognition. Although Google Scholar is the most prevalent literature resource for this thesis, other notable engines such as “Core” and “Scopus” were also utilised for various papers. Other prominent resources such as the Vrije Universiteit library, ResearchGate, Springer and ArXiv were commonly checked for additional material on these topics. The main keywords for searching the current literature are “named entity recognition”, “ner”, “anonymisation”, “pseudonymisation”, “de-identification”, “dutch”, “big data” and “privacy”. When selecting papers, both the publishing date and number of citations by other papers were taken into consideration. Moreover, the popularity of a particular task or subject is considered. For instance, NER in general is a relatively popular task and many papers were written for this topic. In this scenario, the relevance of both date and number of citations is large when selecting literature. However, papers that discuss Named Entity Recognition on Dutch documents occur much less frequently and hence the number of citations and publishing date becomes much less essential when selecting literature. It should also be noted that more recent papers and surveys take priority over older literature. When a relatively new paper is published, only papers after this publishing date can cite these papers and thus the number of possible citations becomes marginally smaller. Thus, this would give the older papers an unfair advantage if this was not taken into account. Moreover, the most recent papers generally attempt to use the best techniques at their disposal and thus more incentive is given to prioritise the newer papers over their older variants.

To better comprehend the scope, popularity and diversity of the chosen research topics and papers, various charts have been created based on the data from the literature engine “Scopus”. Figure 1 shows the main trend of the term “Named Entity Recognition”, while Figure 2 shows the trend of de-identification-based terms. These documents were obtained by specifically searching for “Named entity Recognition (Dutch/English)” within the abstracts, keywords and titles of numerous papers. Although Scopus does not necessarily represent all the literature covered in this study or on other

search engines, such as Google Scholar, it does provide a relatively reliable impression of the popularity of the various research topics and is supported by the current literature in this thesis. From Figures 1 and 2, it can be deduced that the de-identification-based terms provide the earliest papers. The first paper for these terms, according to Scopus, was published in 1947, while the first paper for Named entity recognition appeared in 1984. However, both terms started to gain popularity in the the beginning of the 2000's and gained notably more traction around the year 2016. This is most likely related to the new GDPR law that was being worked on around this year, while coming into effect two years later. In addition, Figure 1 shows that English and Dutch languages contribute relatively little to the overall research of named entity recognition. This is supported by Nadeau and Sekine (2007), since research for NER seems to revolve around multiple languages rather than only English. However, it is also likely a consequence of some papers simply not mentioning the term "English" in their papers. Moreover, given these current trends, it appears that neither literature topic will decrease in interest in the foreseeable future.

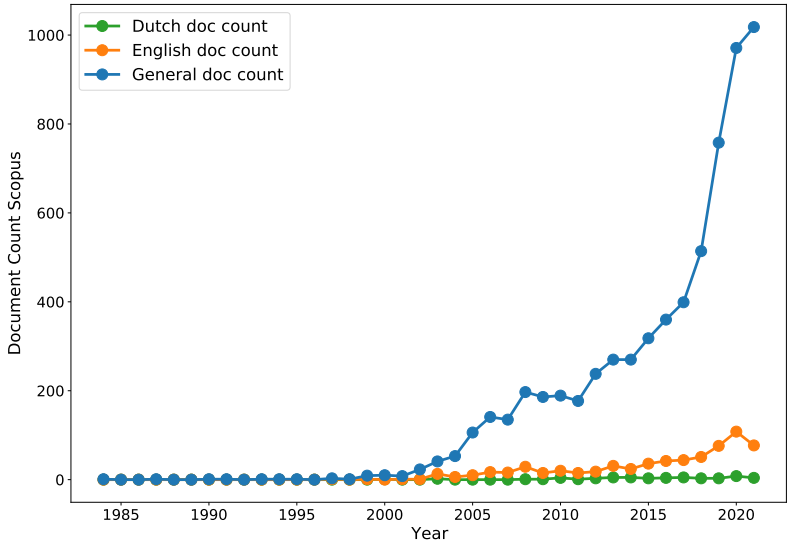

Figure 1: Trend of "Named AND Entity AND Recognition (AND Dutch/English)" when searching for these terms within the keywords, abstracts and titles of numerous literature documents for Scopus. The "general" count represents the number of documents without any language filter.

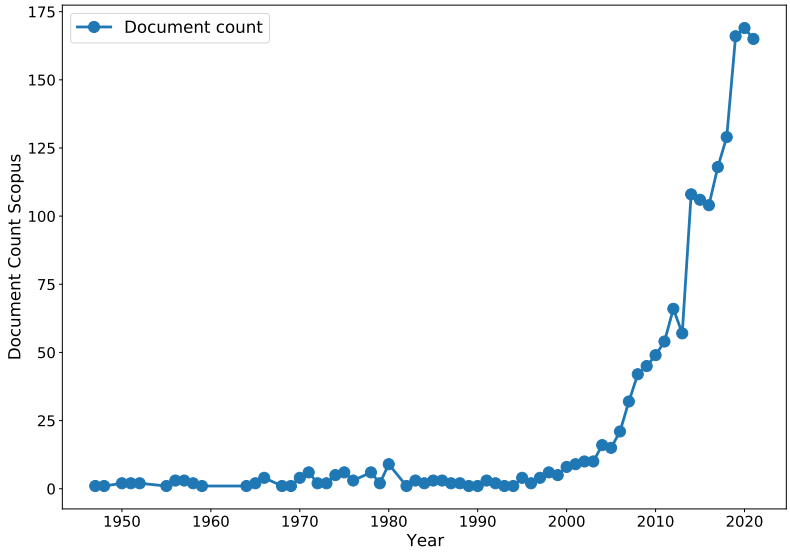

Figure 2: Trend of "De-identification OR Anonymisation OR Pseudonymisation" when searching for these terms within the keywords, abstracts and titles of literature documents within Scopus.

In addition to the previously discussed trends, the subject areas and document types were also checked for both the de-identification and named entity recognition searches. The bar chart in Figure 3 depicts the different subject areas for the de-identification-based term searches. The main reason for showing this graph is to indicate the diversity of the subjects when considering de-identification-based terms. The most notable detail in this visualisation is that the field of *Medicine* is the second most frequent subject area for de-identification literature, which may come as a surprise at first glance. However, the medical domain contains some of the most privacy-sensitive data and would thus explain the popularity of de-identification-based research projects. This is supported by the literature in this research, since many de-identification papers use the medical domain as the basis for their research. This is also true for the Dutch language, for which a paper was written by Menger et al. (2018) for instance. Moreover, the *Other* category also indicates that there is a large variety of fields in which de-identification is a research topic. When it comes to Named entity recognition, the field of medicine becomes much less popular and the most prominent areas are Computer Science and Mathematics. Since this is much less relevant, this graph can be found in the Appendix. Moreover, these searches also show that the most prominent document for de-identification papers is *Articles* while the most used document type for named entity recognition is *Conference*. As will be shown later in the NER literature section, this is possibly due to the introduction of NER through many conferences and tasks. Moreover, it is also likely that creating tasks or measuring performance is much more difficult for de-identification-based scenario's in comparison to NER given the complexity of visualising and proofing the performance de-identification-based methods.

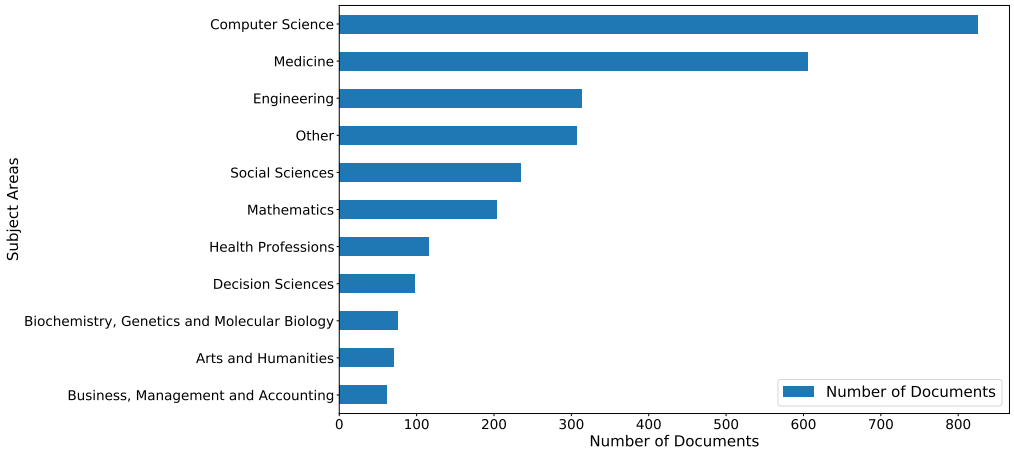

Figure 3: The different subject types for the search "De-identification OR Anonymisation OR Pseudonymisation" on Scopus.

2.2 NER Definition

The definition for named entity recognition can be split into 2 separate parts: "named entity" and "recognition". The "named entity" part originates from the rigid designers defined by Kripke (1982). The authors stated that only proper names and natural kind terms (e.g., substances) belong to the task. However, in practice, this definition is loosened for practical reasons, as was indicated by Nadeau and Sekine (2007). These designers are the components of a sentence that are being seen separately from this sentence. To help illustrate this concept, words and tokens that are critical to the meaning of a sentence, such as persons and organisations, are considered named entities, while stop words and punctuation are generally not considered named entities. "Recognition", in this paper, refers to the automatic localisation and classification of the named entities in the text using Natural Language Processing techniques. It should also be noted that this recognition can be applied to both structured and unstructured text. However, the documents that are processed by Stater have not been pre-processed or structured yet. With this information in mind, the definition that will be used for Named Entity Recognition in this thesis is as follows: Named Entity Recognition (NER) is the process of automatically localising and classifying pre-defined rigid designers in unstructured text.

The definition of NER can also be defined in a more mathematical and practical way, as was shown in the survey by Li et al. (2020). Therefore, this paper will utilise this definition, al-

though some terms will be renamed for better clarity and consistency. Given a sequence of tokens $T := t_1, t_2, t_3, \ldots, t_N$, where *N* is the length of the sentence in tokens, NER tries to find the token sequences $S := s_1, s_2, s_3, \ldots, s_M$, where *M* is the number of sequences found in the sentence, such that each sequence can be tagged according to a pre-defined scheme of labels. Generally, these sequences contain the starting index of the sequence, the ending index of the sequence and the corresponding label. This results in the following definition for an individual sequence:$s_i = (c_s, c_e, l)$, where s_i represents any sequence in *S*, *c* represents the character index in the sentence, c_s and c_e the start and end character position of the sequence and *l* the label assigned to the sequence. This is a format that is often seen in applications that use NER-based functions such as the"SpaCy" library, which will be explained in detail in the Model Descriptions section.

NER consists of two main tasks, namely identifying the location/range of a named entity and the classification of the named entity according to pre-defined categories [26]. To make this distinction more apparent, the recognition of entities is considered a syntactic task, while the classification is considered a semantic task [26]. The task of identifying the location and range of the named entity is considered more meaningful for this project, since the de-identification of an entity takes priority over the context of the text. Despite the importance of the first task, the second task will still be considered for this project, since context is lost if the entity were to be replaced by a generic and fixed string. Moreover, there are numerous different approaches to localising entities, which, if considered by themselves, can not only lose context but also add more confusion to the text. Moreover, pseudonymisation, a de-identification technique, requires the NER label in order to work properly. Therefore, the distinction between these two tasks needs to be accounted for when evaluating the NER results. This is solved by considering two different evaluation metrics, which are described in the Set-up Experiments section.

Li et al. (2020), as was also hinted at by Nadeau and Sekine (2007), mention that named entities can take on numerous distinct definitions and that the research community struggles with a consensus on these definitions. This may be a consequence of named entities in general (92.2%) containing uncommon words, while 50% of named entities have a "loose" structure [26]. What researchers do commonly agree on is that there are two types of named entities, namely generic and domain-specific entities. Generic named entities are entities that occur across various domains, such as *persons* and *locations*, while domain-specific entities refer to entities that rarely occur across different domains. For instance, the medical domain, which is regularly used as a domain for NER tasks, utilises words related to genes and proteins. Some works have further expanded upon these definitions by using a more fine-grained approach to defining the label schemes. This can either mean creating many more specific labels for the generic entity types, where a *politician* is part of the generic type *person* [22], and/or where each individual token in the sequence gets its own label [23]. However, such fine-grained definitions will increase the complexity of the task, which will thus not be discussed in this thesis given high complexity of the current task and will thus be left to future work.

2.3 De-identification research

For this thesis, the concepts of anonymisation, pseudonymisation and de-identification will be used regularly. These three terms will be properly defined in this section to avoid confusion. Anonymisation, as defined by the GDPR, concerns "information which does not relate to an identified or identifiable natural person or to personal data rendered anonymous in such a manner that the data subject is not or no longer identifiable" [11]. Pseudonymisation is defined as "the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person." [27]. As identified by the Spanish Data Protection Agency [28], a critical distinction between these two concepts is that, with additional information, the data subject can be re-identified with pseudonymisation, while for anonymisation this ought to be completely impossible [29]. In addition, while the former can retain some of the original quality of the data, the latter guarantees better privacy protection. Both anonymisation and pseudonymisation fall under the umbrella of de-identification, which is a "general term for any process of reducing the association between a set of identifying data and the data subject" [30].

Since the introduction of the GDPR and other data regulation laws, such as the Open Data Directive [31], the concept of de-identification has seen a large spike in interest within the past few years, as was shown in Figure 2. Due to many countries developing their own regulations with regards to

data privacy [24] and due to the introduction of Big Data [32], the process of de-identification has become extremely complex. Additionally, the process of de-identification is costly and easily prone to errors if this tasks is performed solely by humans [33, 24]. As a result, AI-based models have been used as the main solution for de-identifying data. It is also expected that these models will grow in popularity in the coming years for more domains, such as the legal domain [33]. Furthermore, there are currently no restrictions on the use of specific techniques for anonymisation [33]. Consequently, research is still ongoing on the correctness and practicality of many techniques including NER [33], of which many currently indicate that the expectations are set too high for these types of techniques [32, 34, 35]. Some papers claim that anonymous data in general does not ensure privacy [34] or that anonymous data can never ensure privacy for big data [32]. This is mostly due to most NER techniques not being able to fully anonymise information in the text such that it conforms to the GDPR regulation [11]. As was mentioned previously in the anonymisation definition, individuals or organisations should not be able to be identified at all, even if extra information would be made available for this process. However, this is extremely difficult to accomplish. To illustrate this complexity, one study tried to re-identify anonymised text and succeeded with this re-identification with an accuracy of 85% [36].

Despite these sceptical responses to current NER research for anonymisation tasks, some research demonstrated great successes in the field of NER and have shown that automatised techniques and frameworks can outperform manual anonymisation of documents [37, 24, 38]. Additionally, even if NER by itself is not sufficient for anonymisation, it is still considered a critical component for correctly anonymising these documents [35, 34]. Moreover, it has been shown that pseudonymised data can be judged by external individuals as anonymous [39], which makes this issue even more complex. Thus, the most critical aspect of de-identification may be the relationship and communication between all involved parties [39]. Although this perception exists about pseudonymisation, it is overall less capable of protecting data in comparison to anonymisation, as can be identified through their difference in definition. Although pseudonymisation is less strict in legal terms, which thus makes the data more privacy sensitive, there are still advantages to applying this technique to sensitive data. For instance, it ensures better data security, integrity and quality [33] if applied correctly, which is always a better option than leaving this data be.

Despite these positives, a trade-off always needs to be made between utility and privacy when it comes to anonymisation [33, 40, 41, 16]. The more the data is anonymised, the less useful it becomes overall when considering it from a data science perspective. In contrast, from a legal or customer perspective, the data needs to be anonymised thoroughly as to not break laws or break the trust between the customers and researchers [24]. Therefore, it is crucial to determine the use case of every de-identification-based task. In addition, it is equally important to have a clear list of personal identifiers that need to be anonymised or pseudonymised for the task of de-identifying information inside documents. Some of the previously referenced papers provide such guidelines for both the medical and human resource domain [24, 35]. Hence, for this research, the legal department at Stater will be involved in the research of this thesis. This will be discussed further in the Annotation Process section.

2.4 NER History and Evolution

As is detailed in various surveys [22, 42, 21, 23], the task of Named Entity Recognition was first introduced during the Sixth Message Understanding Conference (MUC-6) in 1996 [43]. Before this conference, the task and term of NER was relatively unknown, but its importance became clear when it was introduced as a subtask of the Information Extraction (IE) field. These tasks showed the importance of locating and classifying information units in unstructured text and the task became known as Named Entity Recognition and Classification [22]. This thesis will simply refer to this task as Named Entity Recognition (NER), since this is a term frequently used in recent literature. After the MUC-6 conference, the interest in NER tasks increased and various other scientific conferences and NER-based tasks were created as a result, such as CoNLL03 [44], ACE [45] and TREC [46]. Although this thesis mostly focuses on the de-identification of financial documents data, NER has seen usage in a variety of applications, such as question answering [47], information retrieval [48], topic modelling [49], automatic text summarisation [50] and machine translation [51].

In the beginning of the NER tasks, rule-based and handcrafted systems were used to perform this technique. These systems rely on grammars and lexicons to extract entities from unstructured text [52, 53]. However, due to the lack of generalisability of these models, it is difficult to transfer these

techniques to other languages, domains and tasks[52]. With the rise of machine learning algorithms in the early 2000's, a combination of supervised and unsupervised techniques became the most prevalent method for solving NER tasks. Such models are able to adapt to new data relatively well, increasing their precision and thus improve on the performance of rule-based systems. Rule-based systems and (un)supervised learning methods can be combined with numerous models as was demonstrated by multiple papers [54, 21]. Recently, advances in deep learning methods have resulted in further improvements in terms of generalisability, accessibility and results. This gave rise to two of the recently mentioned surveys by Yadav and Bethard (2019) and Li et al. (2020). Each of these developments and methodologies will be described in further detail in the upcoming sections. A small overview of these sections is given in the Appendix in Table 9. Note that not all papers mentioned in the upcoming sections are mentioned in this table, since their evaluation metrics were different in comparison to that of other models. In exchange, extra papers are added to this table to compensate.

2.4.1 Rule-Based Systems

The handcrafted and rule-based systems are mostly based on gazetteers, dictionaries and syntactic lexical patterns. Such methods generally require domain knowledge and thus cannot be generalised well, as was demonstrated by Farmakiotou et al. (2000) and Segura-Bedmar et al. (2013). Some fairly well-known NER rules and systems include the Brill rule [55], NetOwl [56] and LTG [57]. As both Yadav and Bethard (2019) and Li et al. (2020) note, due to these domain-specific rules, these systems generally show high precision and low recall. This is due to such rules always finding entities that are contained in the given dictionaries or rules, while missing out on all entities not included by these resources. Given current research, handcrafted and rule-based systems have seen much less usage of the years when considering them as a model by themselves [22, 21, 23]. This is mostly due to these systems being outperformed by other more complex systems. Additionally, such systems often require high system engineering costs [22]. These systems are thus instead combined with unsupervised, supervised and deep learning models to create hybrid models [54, 23]. As will be detailed later, this generally implies that they are used as feature for the more complex models to help with the detection and classification of named entities. As a result, few recent papers have been written for specifically these systems. Only if there are no training examples available, then rule-based systems are preferred over the other types of machine learning algorithms described in this study [22]. One exception to this trend is a relatively popular Dutch paper that tries to de-identify information in medical nursing notes [24], which will be discussed later.

2.4.2 Unsupervised systems

The main concept behind unsupervised systems is to perform tasks on unlabelled data. This implies that the model receives no feedback from its decisions [54, 23]. Instead of learning from the data using feedback loops or fixed rules, the goal of these systems is to learn representations, such that, through these representations, named entities can be found [54]. Unsupervised learning in general relies on statistics from unannotated corpora, lexical resources and patterns to learn these representations [22]. Additionally, it differs from supervised learning, since unsupervised learning does not require a large and robust number of features and does not need any annotated corpora to work [42]. Examples of such unsupervised systems are *WordNet* [58], Hearst patterns [59] and systems containing correlations between similar sources [60].

Due to the usefulness of such rules, patterns and statistics, they are often used as a basis for many systems and approaches, such as in Cimiano and Völker (2005) and Etzioni et al. (2005), of which the latter is often mentioned by various surveys [63, 23, 42]. It also uses another frequently referenced system which is referred to as the Brill tagger [55], which uses the Brill rule previously mentioned. A more recent paper by Zhang and Elhadad (2013) discusses an unsupervised method for extracting named entities in biological and clinical texts. The authors make use of unsupervised "seed knowledge" and classification and used other papers as inspiration for their own system [58, 65]. The idea of seeds in unsupervised learning is that they represent a set of rules through which information is extracted if this information follows 1 or more of these rules. Through this generation, researchers are able to find and create more seeds from the previous generation. This process is repeated until their criteria is satisfied.

These papers and examples provide ample evidence that unsupervised systems are more easily generalisable to other languages in comparison to the previously discussed rule-based systems due to

them using general patterns and representations [54]. Another advantage of such systems is that they do not require annotated corpora and thus more time can be spend on obtaining an adequate model rather than annotating large corpora. Although more papers are found for unsupervised systems in comparison to pure rule-based systems, these systems also do not seem to be as popular as the supervised and deep learning alternatives [54]. If these systems are considered, similar to the rule-based systems, they are mostly used as features in supervised or deep learning techniques and models, since the latter models in general outperform the rule-based and unsupervised approaches.

2.4.3 Supervised Systems

Supervised systems are architectures that use labelled corpora for learning specific tasks and apply this knowledge on unseen data [23]. Due to the fact that supervised and deep learning models need labelled data and robust features and/or representations, both techniques do cost the most time and effort in terms of creation and initial effort in comparison to the previously discussed techniques. Unlike deep learning systems, supervised learning methods generally require more features for them to work as intended and learn the task of NER. This is due to their sensitiveness to feature sets and due to the complexity of languages in general, as was shown and concluded by Tkachenko and Simanovsky (2012). However, once the training data is acquired and the corresponding features are created, training supervised models is more effective for learning the NER task in comparison to the previously discussed methods. As Nadeau and Sekine (2007) mentioned in their survey in 2007, it was the dominant model at the conference CONLL-2003 and is still an influential model nowadays, although its usage has been lessened due to the introduction of deep learning-based methods. Given the importance of feature engineering for this system, such information and techniques will be discussed first. Afterwards, some models will be discussed and explained in further detail.

Features

Given the large amount of information in text, there is an enormous number of features that could be engineered. As is indicated or hinted at in various NER surveys [22, 42, 21, 23], all these features can be split into various categories. Generally speaking, there are 3 types of features: word-level, list lookup (dictionary-based) and document-/corpus-level features. Although word representations and embeddings can be used for supervised models as well, this type of feature will be explained later in the Deep Learning section, since they are the most important for these types of models.

Word-level features concern themselves with providing the model information on the word itself. There is a wide variety of features for this category, such as case-dependent features, punctuation, digits, morphology and part-of-speech (POS) features [22]. Although such features are the simplest to extract, they can be crucial for detecting entities. For instance, capitalisation is one of the major methods for detecting named entities in text, as names are often indicated by a capital letter in the beginning of a phrase or word. Additionally, certain digit patterns can be detected with these features, as was proven by Bikel et al. (1998) and Yu et al. (1998). Moreover, several functions can be created to extract more advanced features (non-alphabetic versions of words) or create new pattern features (“A-A-‘’ or “Aaaaa-000”) and was introduced by Collins (2002). As was shown by Chiu and Nichols (2016), using such information in combination with supervised or deep-learning models gave promising results.

The second type of feature to be considered is the list lookup feature, which is often used interchangeably with dictionary, gazetteer and lexicon, although there are slight differences. These differences will be explained here to ensure consistent definitions throughout this thesis. A dictionary generally references a work where words from one or more languages are explained and the origins of these words are detailed. A gazetteer is an encyclopedia or geographic dictionary, which is thus a form of dictionary. A lexicon is the vocabulary of a language or document. The main concept behind these features is to use general lists of entities, cues and other word types to provide models with more information about the meaning or context of specific words. For instance, it could be helpful for the model to know that “Amsterdam” is a city in the Netherlands. These dictionaries are useful for disambiguation of nouns, as was shown in [71], where both these dictionaries and the positions of nous helped disambiguate these words. They are also used to identify words frequently used in organisation names [72, 73]. However, such methods do generally require exact matches. This creates problems given the high variety of wording and potentially incorrect spelling in the provided unstructured documents. To help with such problems, stemming or lemmatisation [74], fuzzy-matching [75] or the Soundex algorithm [76] can be applied. Despite the usefulness of such

features, it should be also mentioned that such features require upkeep, which can increase costs as well, which is important to consider when using these dictionary-based features.

The third type of feature is the document- or corpus-based feature which tries to collect information from large collections of documents. Examples of such features include occurrence-based information, local syntax, meta information and the frequency of words or tokens [71, 77]. Mikheev (1999) and Thielen (1995) used this information to check both lower- and upper-cased versions of the token to help the disambiguation process. Additionally, such information can help find co-references (references to given entities [22]) and aliases of the current token/word. As was shown by Zhu et al. (2005), meta-information can help bias the probabilities of certain words towards specific labels. For instance, email headers can help with identifying persons, since emails generally contain names of persons [22]. This helps the NER task, since word-level features are often insufficient for solving these complex problems, as was shown for semantic tagging Poibeau (2006). Some other notable context or document related features include window size of context/neighbouring words [80] and the unsupervised clustering of corpus features (Brown corpus in this example) [81].

Models
Due to the large increase in popularity of machine learning models over the years, there is a wide variety of possible models that can be applied for this task. Based on the results from various surveys [22, 42, 21, 23], the main models that are used within this field seem to be Hidden Markov models (HMM), Decision Trees (including variants such as Adaboost), Maximum Entropy models (ME) and Conditional Random Fields (CRF). HMM, decision trees and ME models seem to perform the worst out of these possible supervised models given these same surveys. The most popular of these supervised models currently is the Conditional Random Field, which is now often used in combination with deep learning models such as the BI-LSTM to obtain some of the highest performances for the task of NER [23]. Experiments were performed with these models and were mostly used in the early 2000's, as is indicated within these same surveys. Although most of these models have seen usage and improvement in recent years, mostly in combination with more complex models, the Maximum Entropy model has not seen much attention. From these, models, the CRF will be used in combination with the BERT model for the NER predictions in this thesis and will thus be elaborated on in the Model Descriptions section.

2.4.4 Deep Learning Systems

Two of the previously published NER surveys [21, 23] discuss the latest developments of deep NER systems. Due to the advancement of technology, more complex models have been introduced in the machine learning field, such as the Feedforward Neural Networks (NN) and Convolutional Neural Networks (CNN). These models have taken the machine learning field by storm and are responsible for large improvements across a variety of fields, such as Computer Vision and NLP. NLP has mostly seen large improvements from the introduction of the CNNs and more recently the Bidirectional Encoder Representations from Transformers (BERT) by Google AI [6]. The latter model has been especially successful for language, in comparison to previous models such as the Bi-directional Long Short-Term Memory model (BLSTM). This is mostly due to two reasons. First, BERT is able to learn in an omni-directional manner, which helps BERT to understand the complexity of language better. Second, the model is parallelisable, which enables BERT to learn much faster and thus learn on much more data [6]. The workings of this model are fully described later in the Model Descriptions section of this thesis. However, as is indicated by various papers, the success of Deep Learning systems generally comes from a combination of multiple models to create one successful model. This is most likely a consequence of NER being considered a combination of two different tasks, which was also indicated in the NER Definition subsection. Moreover, in order for these models to work, language needs to be represented differently. To make a distinction between these techniques, for simplicity, this thesis will first discuss the embedding/representation techniques and afterwards discuss the architectures corresponding to these embeddings.

Word representations
There are 3 types of word representations identified in the literature, namely word-level, character-level and hybrid representations [21, 23]. Word-level representations imply that the words in a sentence are encoded. These representations are generally pre-trained through unsupervised algorithms, such as CBOW and skip-gram [82], which are both used in the Word2Vec model [83]. Some of the most popular methods for this type of representation are GloVe [84] and fastText [85]. Character-

based representations are created by using the individual characters as input into a neural network based model, such as CNNs [84] and LSTMs [86]. After the predictions, the character labels are then post-processed into word labels to conform to the testing dataset by, for instance, using a Viterbi decoder [86], which showed great generalisability for multiple languages [21]. This generalisation originates from the learning of characters, which manages to better adapt to new data. Additionally, character-based information, such as prefixes and suffixes, are also learned well through this technique [23].

However, in comparison to each individual method, most papers try to obtain hybrid representations of sentences, as is also shown by the same two surveys mentioned previously [23, 21]. Not only do these representations include a combination of word-level and character-level representations, but also include various supervised features mentioned previously. They are considered strong systems that need little to no domain resources [21] if only embeddings are used for these models. However, they are able to include supervised features through extra layers, which helps to enhance the capabilities of the NER models. The most frequently used features include capitalised features [87, 70] and lexical features [88, 89]. Additionally, Yadav et al. (2018) showed that affixal features improved performance, which obtained the best F1 score for the Dutch language in their survey. The overall scores also improved in comparison to the individual types of representations [84].

Models

The earliest architectures that were used for the task of NER were mostly Feedforward Neural Networks [91] and CNNs [92]. However, these types of models were replaced by the BI-LSTMs, which are able to capture both future and past information much more successfully and naturally. Although the CNN is much easier to train due to its parallelisation, this BI-LSTM network is generally able to obtain better results and thus became the "de facto standard" [23, 84, 93]. However, for most of these models, an extra layer was used on top of these networks, such as a softmax [94] or Conditional Random Field (CRF) layer [88] to improve performance. Additionally, it is not uncommon that multiple models are combined into one architecture, as was done by Limsopatham and Collier (2016) and Žukov Gregorič et al. (2018). This is most likely done for contextual purposes, since the representations work on word- or character-level and not on sentence-level. Other notable yet less popular methods include Recursive Networks [97] and Neural Language models [98]. The former is not mentioned much in the literature, while the latter is often used as feature in other networks, such as by Peters et al. (2018). The most recent major development in this field was the introduction of Transformers [3], which led to the models of ELMo [99] and BERT [6]. These models are generally used for encoder representations [23].

2.5 Dutch NER systems

Although English is the most prevalent language when it comes to the NER task, multiple other languages also have been studied, such as German, Spanish and Dutch [22], as is supported by Figure 1. This section will focus on the Dutch NER systems, since the objective of this thesis is to deal with Dutch documents. As a result, only few papers will be discussed, due to the relative unpopularity of the Dutch language on an international level. The paper by Meulder et al. (2002) mostly focused on combining rule-based systems and statistical systems to detect named entities in Dutch financial newspapers. They used a total of 7 different gazetteers of place names, company names and first names in addition to name grammars. They note that Dutch names often contain prepositions with lower cases, which makes detecting names in Dutch a bit harder than in English, although these problems can still be solved by applying some extra grammar-based rules. Additionally, they make a distinction between person and company names as well as surnames in their architecture and for their rules.

The main unsupervised system used in this article is the Ripper rule-learning model by Cohen (1995), which was mostly used for context-based rules. Even though it was state-of-the-art at the time, their main shortcomings included the flexibility of the system and the narrow scope of the context rules. This performance was improved by another early paper on multi-lingual NER by Carreras et al. (2002), which won the CoNLL 2002 conference. Their overall performance was at least 6% better than that of Meulder et al. (2002). They used binary and domain-dependent features, such as capitalisation, trigger words and gazetteers and ported their initial model in Spanish to the Dutch model. However, due to these domain specific features, flexibility is still being hampered. In contrast, a more recent paper by Agerri and Rigau (2016) used a combination of orthography,

n-gram and unsupervised features, which achieved near state-of-the-art performance on a Dutch corpus. Therefore, even without using deep neural networks, the performance and generalisability of these supervised feature-engineered systems is respectable.

As was previously mentioned, one relatively well-known paper for Dutch NER is the article by Menger et al. (2018). In this paper, a variety of rules have been used to detect various PHI (Protected Health Information) categories, such as person names, geographical locations and patient numbers. These categories have been created with the help of the local medical staff to help with identifying the most crucial information in the text. To summarise their rules and patterns, person names are identified by assuming a sequence of alphabetic characters by non-alphabetic characters excluding specific prepositions in Dutch, such as "van der". Most of their information on many of these categories, such as persons and institutions, originated from internal systems, while most geographic locations were obtained through multiple lexicons (one for European cities, the other specifically for all Dutch villages and cities). For geographic locations, Dutch postal codes adhere to clear formats and have few variations, namely four number followed by two letters. Similar patterns and rules have been used for patient numbers, telephone numbers, dates and ages.

However, with the introduction and popularisation of neural networks for supervised tasks, most papers now employ these more complex models for various NLP tasks. The most crucial and influential system that was introduced is "BERTje", which is a monolingual Dutch version of the BERT model that was introduced previously [103]. Their pre-training of a monolingual word-level Dutch version of BERT outperformed the multi-lingual word-level variant [103]. Another model that tries to improve upon this model is the "RobBERT" model, which uses the larger version of BERT and more extensive preprocessing to create their own state-of-the-art model in various tasks. Given their great performance in general and due to their architectural and learning differences, both will be used within this thesis and are described in more detail in the Model Descriptions section. The NER survey by Li et al. (2020) showed that using BERT as language encoders outperformed all other models for the NER task. Additionally, Feng et al. (2018) showed that, combining the previously most popular NLP model (Bi-LSTM) with a CRF layer and various lexicons, achieves a F1-score of 88.39% on the Dutch language, which was the highest score obtained in this paper in comparison to the score of 86.82% for Spanish and 83.07% for Chinese. This is the main reason for choosing the BERT models for the experiments within this thesis. Yadav et al. (2018) also showed great results on the same CoNLL dataset, although it is lower by 0.5% for the Dutch language. They used word and character representations, in combination with an affixal features and RNN models to perform the NER task. Despite these great results, it should be noted that these authors looked at words that both did and did not occur in the test set and showed that, for new words, the F1-score is about 30% lower than that of the words that occurred previously (97.00% in comparison to 64.10%). This is important to investigate further when evaluating the performance of the models in this thesis.

3 Data Exploration and Selection

There are two datasets that were provided for this research. The first dataset that will be investigated are email texts from the department Document Management (DocMan for short). This department is responsible for controlling, scanning and assigning tasks to emails that are received by Stater. The second dataset contains documents that were send to Stater's customers or Stater themselves for various inquiries. These documents are stored by an external company that works closely together with Stater. The origins and statistics of these datasets will be described below. Next, a document selection process is performed, where specific documents will be selected for both training and testing the models. Afterwards, the datasets are split into train and test datasets, where documents from both datasets will be chosen for their respective set and combined afterwards for the annotation process. The annotation and post-annotation processing of these documents will be described in the Annotation Process and Set-up Experiments sections.

3.1 E-mail Dataset Description

This section will provide more information on the e-mail dataset. Although this dataset is not considered as crucial for de-identifying information to Stater, there is a large amount of text contained in this data with a large amount of privacy sensitive information. Therefore, this dataset is crucial for both training and generalising the various models that will be tested in this thesis. Moreover, this

data was used previously for a different project, which made it easily accessible. To understand the origins of the data better, first the corresponding department's process will be discussed. Afterwards, more details will be provided on these files in the Data Exploration section. For confidentiality reasons, some names will be de-identified in these sections.

3.1.1 DocMan general process

The DocMan process can be divided into several steps. First, when an email is being sent to Stater, it is processed by the Document Management (DocMan) department. This department checks if all the required information is present within this document, such as the name of the customer, the customer loan number and any corresponding documents for a given request. The customer name and loan number are checked by locating and cross-validating them manually with the provided documents. An email is returned to the sender if there are any discrepancies during this check. Second, the corresponding task numbers are assigned manually to each incoming mail. This is referred to as "indexing" within the company and this term will be frequently used throughout this thesis. The main goal of assigning these numbers to the emails is to identify each email as a certain task that needs to be processed by the other departments or systems connected to Stater. Another goal of this task assignment process is to provide the receiving department with additional information on the subject matter, such as the name of the customer and their corresponding loan number, etc. Third, these texts are send to the corresponding departments and systems for further processing their content and information.

Most of these emails will be received in the standard email environment. However, there are separate environments that are set up for specific customers. For confidentiality reasons, the names of both the customers and consequently these environments will not be revealed. In addition to these environments, recent developments have changed the process of indexing these emails. This improvement was handled by the Datalab and concerns the topic of Email Routing. In short, the assignment of tasks to emails is now partially automated by making use of machine learning algorithms. This was the project for which this email dataset was also used. However, despite the difference in project, the quality of the project should not be hampered, since only a subset of the original data was used for the other project. To ensure there is no loss in data quality, all of the initial emails will receive a general re-examination and a detailed selection process will be carried out for each of these documents. Moreover, some assets will be re-examined and adjusted for this project, which helped speed up this process.

3.1.2 Email Data Exploration

Data overview

To obtain a better understanding of the email dataset, the structure of the folders inside this dataset will be detailed. The main folder contains all the various mailboxes that are used by the *DocMan* department. The names of these directories start with the department name and are followed by the corresponding email environment. Since there are a total of 3 customer-specific environments within this dataset, they will be referenced by their corresponding number for confidentiality reasons. Each of these folders contains two subfolders, which are referenced by "Inbox" and "Sent Items". The "Inbox" folder contains all the emails that were received by the department, while the "Send Items" folder contains all the mails that were then send to the other departments or systems. Although this setup may appear straightforward, there are many discrepancies within and across these subfolders. One of those discrepancies is that one email can concern different tasks for different departments, which results in an extra task being assigned to the same email. This creates a duplicate mail with the only difference being the task number. Moreover, some emails from the Inbox folders are not found in the Sent Items sub-folder and vice versa. This creates inconsistencies between the items in the inbox and the sent item box that need to be dealt with properly when selecting documents for the annotation process.

Figure 4 represents the distribution of all the emails in the dataset for each sub-folder and for each email environment. The names on the upper x-axis of the graph represent the department names, while the names on the bottom x-axis show the different email environments for each of these departments. The "Default" environment represents the main environment used for the mails, while the other environments correspond to the customer-specific environments mentioned previously. As

a result, the “Default” environment is present throughout all departments and thus most emails are contained and received in this environment.

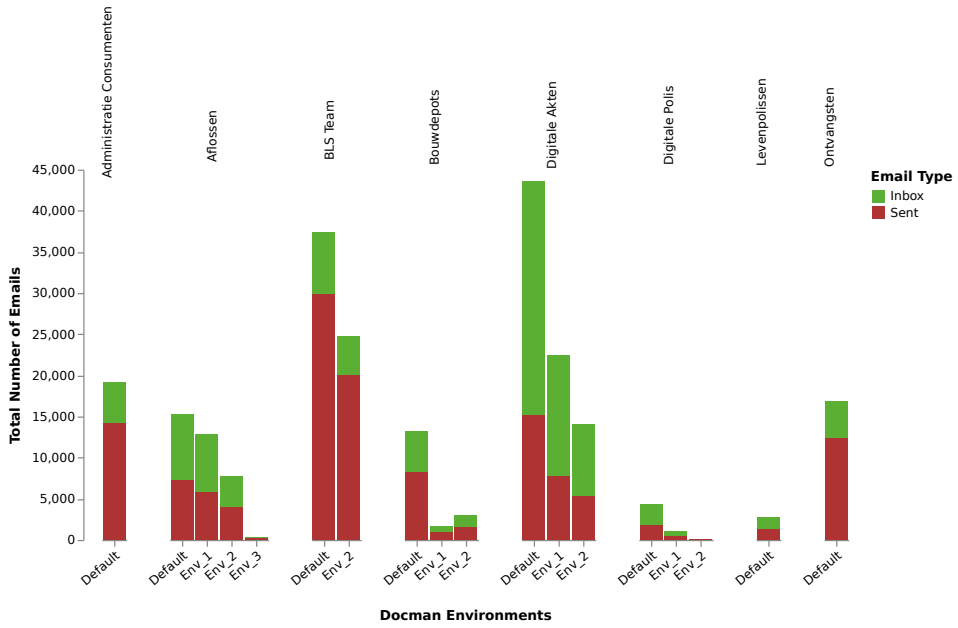

Figure 4: The number of files for each folder in the dataset. The DocMan environments are pseudonyms for the real environments that are used within the company.

There are clear differences in the total number of emails per department. The departments “Aflossen”, “BLS Team” and “Digitale Akten” are the most active email departments, while “Digitale Polis”, “Levenpolissen” and “Bouwdepots” are the least active. This makes sense, since a category such as “Aflossen” (paying off mortgages) is generally more dynamic and influences more departments in comparison to the “Levenpolissen” department (life insurances). As was previously observed and discussed, there are some inconsistencies in the data and some of these inconsistencies are visualised here. The “Digitale Akten” department shows that there are slightly more emails in the Inbox than in the Sent Items, while the “BLS Team”, “Administratie Consumenten” and “Ontvangsten” contain proportionally more emails in the Sent Items mailbox. The former is generally caused by the misplacement of emails in a given inbox, while the latter is related to the indexing of one email multiple times for different departments. Although this is not reflected as much in this graph, the misplacement of emails can occur for both email types.

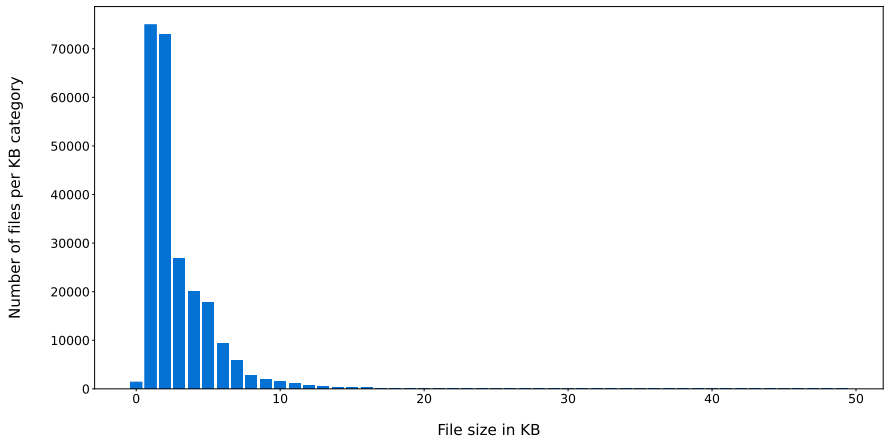

Figure 5: The distribution of the file sizes of all e-mails.

The files themselves vary frequently in size. This distribution of the email sizes is shown in Figure 5, where the number of emails per Kilobyte is depicted. Various probability distributions were checked for the data, but all of them resulted in rejected p-values, which implies is does not follow any standard distribution in particular. The details of this experiment can be found in Table 10 in the Appendix. However, what can be observed is that the files are distributed in such a way that most of these file sizes are relatively small, whereas only a few have a very large file size. Logically speaking, most people will write short emails for time related reasons and hence this makes up the majority of the dataset. This is important when selecting documents for the annotation process, since these models work on a sentence level, which requires having whole sentences as input. In smaller emails, such sentences are, generally speaking, much more rare. Therefore, many of the smallest email files were investigated manually. From this investigation, it could be concluded that these smaller emails indeed do not include much text. However, it was also observed that the file size can be misleading in some cases, since their size can be attributed to either many email headers, much whitespace or large webpage links. Therefore, these factors should also be considered when selecting these documents for evaluation. The methodology behind this selection will be further elaborated in the Document Processing and Selection subsection.

The names of these files are also crucial for the analysis, since these names follow a specific format. All of the filenames start with the date and time on which the email was either received or send. Afterwards, an "unique" identifier is given to each of these emails. These identifiers have proven to be quite useful, since they span all the different departments, environments and email boxes. As a result, if an identifier is given to an email and there is another email with this same identifier, they always appear to be connected in some form or another. Generally, this connection represents an email in the Inbox that was given a task and send afterwards to the corresponding department. However, replies to these threads of emails can also occur and these replies also govern the same key. Therefore, this identification number can be quite useful when removing similar and related documents. However, as was mentioned previously, there will still be mails that contain mostly the same content with different identification numbers. Therefore, an extra similarity check has been implemented to help avoid content that is too similar.

Email structure

Although the content of the emails varies frequently, since this is unstructured data, there is a standard format that these emails adhere to. Since the email structure is crucial for creating rule-based techniques used during the annotation process, the structure of this format is described below.

Each email starts with at least one email header. Every email header contains at least the email address of the receiver and the sender, a date and time at which the email was sent or received and a subject. If there are additional recipients, then an extra field is added (the "CC" field). In addition, some emails carry attachments, which is indicated separately below all of these headers. In addition, some of these emails contain duplicate or extra lines in the header. Another observation is that sometimes the language for these headers changes. For instance, it is noticed that the emails above the indexation content are generally in English, while the other headers are in Dutch. Furthermore, there can be more than one header in emails, due to either threaded replies or the indexation of emails.

After dealing with the first headers of the email, the main content of the email can be processed. This content generally starts with some form of addressal, although this is not always guaranteed, since the content of these emails varies frequently. Sometimes, numerous task numbers and names are shown in the email message, while others contain simple texts or even no messages at all. This results in many emails containing little useful data to train on, since there is a lack in diversity of the emails or a lack of content. If there is some content in the email, then generally this closed off by some form of signature, although this also varies often. This signature can be fairly useful, since many personal details are provided there. After the signatures, some extra information, such as privacy statements or disclaimers can be included. This can result in tricky situations, since these disclaimers can be both in Dutch and English, which could disrupt the model's capabilities. Hence, this is also something that needs to be properly dealt with when selecting and annotating these emails.

3.2 Inquiry Dataset Description

The second dataset that will be considered for this research is the inquiry dataset. This dataset contains a large number of folders and each folder contains PDF documents that are related to various inquiries. These inquiries are stored by an external company, of which the name will not be revealed. This company has provided Stater with a dump of such documents and have allowed Stater to perform analyses on these documents for research purposes. This dataset was created from the various portals within the company, which thus creates much diversity in terms of document contents. The types of inquiries that occur in this data set includes deeds, divorce filings and donation agreements.

To accommodate for the large variety of documents in this dataset, various general statistics are given in Table 1. In this table, the file count, file size per folder and the file size distribution have been described using various statistical measures. From the high standard deviation and low quantiles it can be deduced that there are a large number of folders that only contain a few documents, as is indicated by the large standard deviation and the relatively low value of the three quantiles. A similar deviation is detected for the file sizes inside these folders. To summarise, each of these statistics indicate a heavy and right-tailed distribution, which is similar to the distribution of files shown in Figure 5 and thus can be explained by similar reasoning. Another crucial observation from this table is that the smallest file in the entire dataset is 0 byte, as is seen in the *Min* row of the *File size distribution* column. The reason for this file being 0 byte is that it is corrupted and cannot be opened by any means. Therefore, all files of 0 bytes will be removed during the document selection process.

	File Count per Folder	Total Size per Folder	File size distribution
Count	65	65	32,141
Mean	494.48	110.150 MB	0.223 MB
Std	1880.68	493.576 MB	1.111 MB
Min	1	0.023 MB	0.000 MB
25% Quantile	1	0.275 MB	0.002 MB
50% Quantile	5	1.647 MB	0.051 MB
75% Quantile	24	15.087 MB	0.097 MB
Max	12,431	3,833.584 MB	34.250 MB
Total	32,141	7,159.750 MB	7,159.750 MB

Table 1: Statistics and distribution of the inquiry dataset.

There is also a diverse number of document types inside this folder based on the various inquiry types described previously. Examples of these types of documents are bank statements (“bankafschriften”), salary slips (“loonstroken”), bids (“offertes”), donation agreements (“schenkingsovereenkomst”) and taxations (“taxaties”). Moreover, there are different subcategories for each of these document types, which indicate either departments or company names and thus their corresponding names will also not be provided in this thesis. Most of these subcategories contain only a few files. After exploring all of these different directories manually, it is evident that there is more variation for this dataset than the previously described email dataset. This variation is mainly caused by a difference in topics, although the structure for each of the documents inside each category can also change drastically. In addition, it is evident that there are three types of documents in terms of content, namely text-based, table-based or a hybrid of the two. This needs to be taken into consideration when selecting the best PDF-to-text conversion tool, since different tools or parameters may help better process one type over the other. Since not all of these directories are relevant for this research, the content of these inquiries will be further discussed during the Document Processing and Selection section.

3.3 Document Processing and Selection

Since the annotation process is a long and tedious process and due to the large variety of content in both datasets, only the documents with the most potential need to be considered. This will decrease the time that is needed to properly annotate these texts and choose the most useful documents. Moreover, the data quality is considered crucial for properly training the models. As was noted by the SpaCy developers, the main reason for NER in general having not perfect accuracy is (mostly) due to the imperfect quality of the data during their testing [2]. Another reason for selecting specific documents for the training of NER models is to not obtain too many similar sentences within both the train and test data and thus help better train and evaluate the models on generalisability. Moreover, all of the labels considered for the training process need to occur frequently, since this could otherwise lead to phenomena such as "catastrophic forgetting". This term implies that some entities will simply be forgotten during the training process due to these entities not occurring frequently enough in the data. Due to the distinctness of the two provided datasets, the document selection for each of these datasets is done differently and thus is described in two different sections below.

3.3.1 Emails

For the selection of emails, all email categories are considered equally important for this thesis and thus all are automatically considered for the selection. There are two key aspects of the emails that have to be considered when selecting specific documents, namely the content and length of the emails. Since there is a large amount of unimportant and duplicate content in these emails, such as the email headers, a regular expression (Regex) system was created for this purpose. Note that this is a different Regex system than what is used for the results. This Regex system attempts to remove any email headers, identification sections, disclaimers, large links and more during the similarity and document length checks. This information is not considered critical for the annotation process, since either they can be easily processed using rule-based methods or they will not be considered for the annotation process due to, for instance, a difference in language or occurring too frequently. In addition, any line breaks, empty lines and multiple whitespace patterns are removed from the document, since these string patterns also contribute to the character count of these strings. Although most of these regular expressions were already made by the Datalab department in a different project, some existing expressions were edited and extra expression were added to the list. A full list of all the regular expression types used for this email selection process can be found in the Appendix in Table 11.

After creating this regular expression system, the documents are selected as follows. First, all the email file names are read into the program. Then, for each of these documents, the regular expression system is used to get a shortened email without email headers, disclaimers, etc. The length of the mail is then checked on the previously extracted data string. If the length of this email is greater than 400 characters, then the email will be considered for the annotation process. The threshold of 400 was chosen, to help filter out any short emails that likely contain too few entities for the NER models to train on. After this check for length, the "unique" email key from the document's filename is checked. If there is only one "unique" key, then nothing needs to be done. When there are multiple emails with the same key, two checks are performed. First, if the email does not contain any form of indexation, this is preferred over the emails that do contain indexation. This is due to indexation not being important for the model to learn. Moreover, these parts can be dealt with by using simple rule-based systems. Due to the previously discussed discrepancy of Inbox items having indexations, simply checking for Inbox or Send Items does not suffice. Therefore, a search is done strictly for the indexation parts of the email, which follow a specific structure. To further bias this preference, the emails that do contain indexation need to be strictly bigger than the emails without indexation. The threshold for this check is set to 100 characters after using the regex system. To illustrate this process in more detail, a pseudocode for this document selection process is created in Pseudocode 1.

However, this does not finalise this process, since it was discovered that there are still many emails that contain exactly or essentially similar in content despite the removal of duplicates using the "unique" keys. The same Regex system is used for this process as well for exact matches and this results in the removal of 5,000 documents. Since there were a large number of documents with essentially the same content, a fuzzy string matching system was implemented. This was done by creating embeddings based on word frequency for each document. After creating such embeddings,

Algorithm 1 Email Selection Algorithm

```
regex_system = compile(patterns)
filenames = read_all_filenames(path_to_directory)
save_dict = empty dictionary
for every filename in filenames do
    file = read_file_from_filename(filename)
    processed_doc = process_file(file, regex_system)
    length_doc = calculate_length_processed_doc(processed_doc)
    if length_doc > 400 then
        if file not in save_dict then
            save file info into dictionary
        else
            old_doc = retrieve_old_doc_from_dict(save_dict)
            has_pattern = check_if_doc_has_signature_pattern(processed_doc)
            old_has_pattern = check_if_doc_has_signature_pattern(old_doc)
            if (has_pattern AND length(processed_doc) > length(old_doc) ) OR
(old_has_pattern AND length(processed_doc) + 100 > length(old_doc) ) OR
length(processed_doc) > length(old_doc)  then
                Replace the old document in dictionary with the new document
            end if
        end if
    end if
end for
```

the cosine similarity score is used to check the similarity between each possible embedding. The cosine similarity score was chosen, since it does not consider Euclidean distance for calculating similarity between embeddings. As a result, the size of the documents is irrelevant for this similarity score, which makes the score only focus on the diversity of the words used within each document. A threshold of 0.80 was chosen, since there are some similar names that occur frequently in every email no matter if the main content is different. Additionally, this still allows to choose between a wide variety of documents for the annotation process. This results in a total of 30,296 emails that will be annotated for the annotation process.

3.3.2 Inquiries

In contrast to the emails, due to the varying content and topics of the inquiry-based data, the directories considered for this thesis were manually chosen and investigated. The reasoning behind choosing each directory is three-fold. First, the data needs to include a variety of entities. If documents in specific directories only include, for instance, money-related numbers, then this directory is not considered useful enough for training the various models in this thesis. This can bias the models too much towards certain entities, which can result in, for instance, "catastrophic forgetting". Second, the entity to sentence ratio needs to be large enough. If a document consists of ten pages and yet only has a few of these sentences with personal and sensitive information, it is not considered as important as documents that contain a lot of private data. Although sentences with no entities are also crucial for properly learning the NER models, this occurs frequently enough regardless of the document type. Moreover, this will significantly increase the speed of the annotation process. Therefore, directories should be filtered based on this criteria. Third, both text-based and table-based data should both be included in the training and testing of the models, since both data types need to be de-identified for this project. Most of these documents can be considered as unstructured, which implies that the models should be able to recognise these entities in a variety of formats and thus both data types are crucial for learning.

With these criteria in mind. a total of 18 directories were chosen out of the 66 for this project, of which 14 are completely unique in terms of topic. The other 48 directories will thus not be considered further given the previously discussed criteria. The unique directories concern the following topics: ownership information, annual reviews, donation agreements, taxations, banking statements, notary documents, offers, salary specifications, building specifications, pay slips and pension overviews. While the overviews, statements, pay slips and offers largely concern table-based data,

some of these documents contain text-based data, such as emails, which contain a lot of useful information for the models. The other types, such as taxations and agreements, are mostly text-based data. These text-based documents also contain the largest individual files of the selected directories. A table with more detailed statistics on the selected directories is provided in the Appendix in Table 12.

After selecting the most useful and important directories from the inquiry dataset, the documents still need to be converted to a text format. This needs to be done, since PDF documents are largely image-based and thus cannot be read automatically by the programs used for this thesis. To solve this predicament, a technique called Optical Character Recognition (OCR) is used, which enables the extraction of text from, for instance, handwritten or image-based text. Various OCR methods and techniques were found and tested for this convergence. First, various Python-related extraction methods were considered and tested. However, many of these methods are outdated, which resulted in many technical difficulties given the current versions used by Stater. Additionally, all of these techniques resulted in a pdf to text conversion where additional line breaks are inserted. This results in text that breaks off whole sentences to keep the original format intact. Since this results in incoherent text, it further complicates the process of properly splitting sentences in large paragraphs of text. Moreover, due to the varying number of line breaks and formats, it is impossible to create a system that could properly segment these sentences and tables.

To remedy this problem, the application "XpdfReader" is used, which is a command-line tool for Windows. This application contains a "pdftotext" tool that can be used through the command line prompt to convert PDFs to text. With this tool, no additional line breaks were added to the text files, which makes it possible to use the "SpaCy sentence splitter" on whole paragraphs of text, while further splitting text on line breaks. Moreover, it was also the quickest method of converting all the files to text files, achieving this in approximately a minute. After some experimentation, the best results were obtained by using the tool on the default mode. Since this mode also worked well for table-based data, this seemed to be the best universal tool and mode to use for the conversion process. The only downside is that it is not as accurate as the previously discussed methods for images within these PDF's. Moreover, problems were noted for some files that contained turned pages. Therefore, there is still text that is not processed correctly and thus such text will not be included in the training and testing of the NER models. Furthermore, the encoding was specifically changed to "UTF-8" to keep the text format between the email and inquiry dataset the same.

After performing this conversion, a second selection process is performed, which is similar to that of the email dataset. However, the main difference is that the regex system is not used for shortening the documents. Instead, only the line breaks, tabs and sequential white-spaces are removed. As was previously discussed, the documents are checked on both length and on similar content using the same thresholds and methodology. Moreover, a cosine similarity score is calculated between each of these documents. For these scores, the threshold was increased to 0.95, since there is relatively less data for the inquiries when compared to the emails. Moreover, if only a few documents are contained in a directory, the files should be kept if they differ a little for testing the generalisability of the model. After executing the same process, the total number of inquiry documents considered for the annotation process is reduced from 8,167 (based on the manual selection process described previously) to a total of 5,793 documents.

3.3.3 Train-Test Split

After selecting and processing these documents, each dataset is split into a train and test dataset. This process is performed before the annotation process to help prevent biases and further complexities after annotating these documents. The documents are split by semi-randomly assigning each document to either the train dataset or test dataset. Additionally, the documents will be shuffled for each dataset to increase the diversity of the documents throughout the annotation process if there are enough documents of each type. Roughly a 1 to 5 ratio is used for splitting the documents into their respective train and test dataset. Since only a relatively small number of files will be used from each train and test dataset, the results of this split will be explained in the Annotation section.

Since it is assumed that all emails are relatively similar in form and content generally speaking, this process can be applied to this dataset without issues. Moreover, each category has more than enough emails, which creates a relatively fair split. However, the same cannot be said for the inquiry dataset, since the documents vary frequently in structure, content and size. As a result, these documents

need to be handled differently. Therefore, a decision was made for each chosen inquiry directory. If the number of documents for a particular directory is smaller than the threshold of 5 (due to the 1:5 ratio), all these documents are added to the test dataset. If the number of documents for this same category is larger than 5, they are split according to the 1 to 5 ratio. This results in some categories, for instance the contract agreement-based documents, being assigned exclusively to the test dataset. This is beneficial for the evaluation, since these documents can be particularly helpful when evaluating the generalisability of the models to new document types. Therefore, for the test dataset, the least frequent documents of the inquiry dataset will be annotated first before moving on to the more frequently occurring documents. When the least frequent documents are annotated, the order is randomised again to prevent any additional biases.

4 Annotation Process

This section will detail all of the information related to the annotation process. To annotate the previously discussed documents properly, it first needs to be determined what data from these documents needs to be annotated in order to properly anonymise and pseudonymise these files. This is discussed and checked with the Datalab, Legal and Compliance department within Stater. Consequently, the annotation schema is created based on the guidelines that were provided by all involved parties and thus this schema conforms as much as possible to both the legal and practical requirements. First, a summary will be given of these guidelines and their consequences to this project will be discussed. Afterwards, the annotation schema will be detailed and its connection to the guidelines outlined. Next, the pre-processing techniques for each of the datasets and the annotation tool will be described. Afterwards, some statistics will be shown for the annotated data and their relevance and impact on the performance of the NER models will be noted.

4.1 Guidelines Personal Information Stater

A document was created by Stater in 2018 that provides details and guidelines on how the company protects personal data and information. Although the information is specifically aimed at technical and organisational requirements for Stater, which will not be further discussed in this thesis, some sections of this document can be utilised for this project. The information that will be used from this document includes definitions of de-identification methods, PIIs (Personal Identifiable Information, referred to as PIDs in the literature) and some relevant design guidelines. Considering the definitions, they are mostly similar to that of the definitions given in the Literature section. However, there is an extra definition for masking: “het vervangen van (een deel van de) data in een veld door een bepaald teken, zodat het risico op misbruik verminderd word”. In short, this implies that the data or part of the data is replaced to reduce the risk of possible abuse of this data. This is demonstrated through an example, where a bank account ID is partially being masked by asterisks. This is basically the same technique that will be used for anonymising documents, although the label itself will be used for masking instead of asterisks.

Another important addition is that randomisation and generalisation is an inherent part of anonymisation task. In the literature, this was done, for instance, using differential privacy and k-anonymity [34]. This implies that such techniques are necessary in order to properly anonymise the documents within Stater. However, given this thesis’ focus on NER specifically, such techniques will be left for future work. The document also provides 3 important terms related to anonymisation, namely “traceability”, “connectivity” and “deducibility”. Traceability refers to the concept that persons can be identified through several records. Connectivity refers to correlations between multiple data sources to re-identify individuals. Deducibility refers to the degree of risk of re-identification. Therefore, if neither of these principles hold for the data, then it can be said it is successfully anonymised, since the individual in question cannot be retraced or re-identified. In practice however, this is extremely difficult to achieve as was shown by [36]. For the exact definitions regarding these topics, they are defined in List 12 in the Appendix.

As for the data guidelines, there are several personal identifiers (PID) provided in this document and they can be divided into 3 different categories: normal, special and sensitive personal data. Name, address and contact information can be seen as the normal personal data, while health care and ID-related information can be seen as special personal information. Moreover, any other special personal information in the AVG should also fall under the umbrella of the special personal data

category, according to this document. The "Burger Service Nummer" (BSN), financial data, company/building data, mortgage data, login details and backlog data fall under the sensitive personal data category. In addition, loan, customer and policy identification numbers are able to identify most of this information and thus should also be identified. This shows the distinction clearly between direct and indirect traceability of information. For Stater, direct traceability of information is understood as data through which it is immediately clear who the person is. Indirect data can be understood as information that does not immediately identify an individual, but can recognise specific characteristics of this individual through which, in combination with other information, the individual can still be identified.

However, according to the TDM (Test Data Manager) guidelines, such numbers or ID have to always be identifiable and thus should never be de-identified fully. Therefore, Stater handles the definition of anonymisation slightly differently in comparison to the GDPR, although Stater made sure that these principles are accounted for. When considering the GDPR point of view, not removing such ID from these documents results in such documents not being fully anonymised. On the other hand, such numbers are critical when searching, evaluating or correcting any information or problems inside the organisation, since customer ID are unique identifiers to individuals. As a result, if these IDs were to be completely lost, Stater can run into various issues or is unable to resolve these issues. Therefore, Stater is allowed to use such ID only if absolutely necessary and this has been written down in the contracts that were signed by the customers of Stater. Given the importance of such ID, it would be beneficial to the organisation if such specific ID can be identified through NER. However, the main issue is that it is extremely difficult to keep ID separate from one another, given that in the datasets external ID are also used for such purposes. This makes differentiating between the different identification numbers much more difficult. Moreover, this would also mean making distinctions between specific ID inside the documents of Stater. For instance, it is difficult to make a distinction between loan ID and customer ID, since they have a relatively similar format. In addition, if such IDs ought to be annotated and correctly identified, this would need to be double checked by the database of Stater, which would take too much time. Due to the previously given reasons, it was decided that these distinctions between the different IDs is not possible for this project and thus these IDs will be combined into one label for the annotation scheme.

4.2 Annotation Scheme

In order for the models to learn which data should be recognised as personal data, it is crucial to define an annotation scheme. The purpose of this scheme is to provide pre-defined labels and use these labels to identify all of the important entities in the data. The complete scheme with corresponding definitions can be found below in Table 2. There are some generic labels that are useful for this project that can be obtained from the literature, such as *person (PER)*, *organisations (ORG)* and *geo-political entity (GPE)*. The creators of SpaCy have defined many more of such labels and thus these labels have also been used as a basis for the creation of this annotation scheme. Although not all instances of these labels are not directly considered private or sensitive information according to the guidelines, they can be intertwined with specific personal information. For instance, it would be extremely difficult for the model to differentiate between date of birth and send/receival date of e-mails. Moreover, date of birth does not occur often enough to justify its own label. Therefore, the decision was made that all date formats will receive the label *DATE*.

Table 2: The annotation scheme used for the annotation process, where the label name, its origins and meaning are described. "Lit." implies that the label originated from the literature.

Label Type	Label Origins	Label Description
CARDINAL	Lit. / SpaCy	Any number that does not belong to any of the other categories. Examples are mobile phone numbers, percentages and pseudonymised tasks.
DATE	Lit. / Spacy	Any entity that can be traced back to dates, such as years, weeks, weekdays, months or any combination of the previous entities.

continues on next page

Table 2: The annotation scheme used for the annotation process, where the label name, its origins and meaning are described. "Lit." implies that the label originated from the literature.

Label Type	Label Origins	Label Description
DEP	New	(Also known as DEPARTMENT or ORG-RELATED) This label includes all names that are related to some part of organisations or companies. This includes the names of departments, systems, products and services.
EMAIL	New	Personal or company-related e-mail addresses.
FILE	New	Any file names that were included as attachments in the e-mails or references to files that contain a file extension, such as PDF, PNG and TXT.
FUNCTION	New	Job-related functions, such as Assistant or Notary
GPE	Lit. / SpaCy	(Otherwise known as a geo-political entity:) Any location that can be identified by a name, such as countries, cities, provinces and states.
ID	Literature/New	Any combination of numbers and letters that form an entity that is directly related to a person or organisation, such as bank account numbers or Stater's customer ID.
MISC	Lit. / SpaCy	(Also known as MISCELLANEOUS) Any entity that cannot be labelled as any of the other labels. This includes spelling/grammar errors, tokenisation errors, concatenations of multiple entities, entities unknown to the annotator, etc.
MONEY	Lit. / SpaCy	Any money-related number, such as transaction values, salaries, etc.
NORP	SpaCy	Nationalities or Religious or Political groups.
ORG	Lit. / SpaCy	(Also known as ORGANISATION) Any name that directly identifies an organisation, fund or company.
POST	New	(Also known as POSTCODE) Any combination of letters and numbers that can be directly correlated to postal codes or boxes.
STREET	New	The names of streets, which include house numbers.
TIME	Lit. / Spacy	All time-related entities, such as hours, minutes, seconds, milliseconds and any combination of these entities. Additionally, any time zone will also be included, such as GMT or words such as "Central European Time"
WEBSITE	New	Any links that can be directly correlated to the Word Wide Web.

However, this does not cover all bases for the definitions provided in the previous section. Therefore some labels are added to this scheme. First, it appears that the *GPE* label does not include street names and postal codes. To compensate, the labels *STREET* and *POSTCODE* are added and handled separately, since they both concern different syntax and occur frequently enough in the datasets. In contrast, no extra labels are added for provinces or districts, since these entities do not occur frequently enough and thus will be combined with the *GPE* label. Second, the label *ID* is added, since the financial sector in general deals with various numerical entities that cannot fall under the same label given their different syntax and degrees of importance, such as loan numbers or customer-id's. The BSN number and bank account number also fall under this category, since there is too little data for either of them to justify a different label. The label *CARD* will include all numerical entities that are not related to ID numbers, such as pseudonymised task numbers and phone numbers. With this addition, a clear distinction is being made in terms of which numbers may be important to Stater and which numbers are not. Third, the label *DEP/ORG-RELATED* is included and will refer to departments, systems or products of organisations. Moreover, due to the relatively frequent occurrence of job functions within the e-mail dataset, a label for job functions is also added called *function*, since this could indirectly identify individuals within organisations. The addition of the labels *e-mail addresses*, *website links* and *attachment names* is due to them also occurring frequently in the e-mail texts. Moreover, they can also contain much personal information, such as person/company names, loan numbers, departments, etc. Since they occur relatively often, it is evident that they each get their own label. However, due to them being easily recognisable using regular expressions, it can be debated whether they should be included when training the NER models, which will be investigated in

later sections. The last label *MISC* is added to ensure that everything not belonging specifically to the previous labels can still be identified. This includes grammar or spelling errors, wrong SpaCy sentence/token splits, etc.

Given the current guidelines provided by Stater and the current definitions defined by the GDPR, this annotation scheme should encompass most of the entities that could both directly and indirectly identify natural persons. However, there will be some data that will not occur frequently enough or not at all within these datasets, which implies that the models cannot train on such data. Login data and passwords will be difficult to detect in this data, since they have not been detected in the data at all. Another category that will most likely not be learned with the current datasets is medical data, since this also does not occur as often. A similar reasoning can be applied to other categories indicated by the GDPR, such as political opinions and philosophical believes even though it is part of the label scheme currently. Also, making a distinction between various identification numbers will be quite difficult due to the extremely similar syntax, as was indicated previously. Solving these problems would either require more data or using techniques that fall beyond the scope of this research. Therefore, these issues will be left to future work. If these labels or types are detected in the text, they will be put into a category that is related to the entity in question or they will be labelled as a *MISC* entity.

4.3 Annotation Preprocessing and Tool

In order to speed up the annotation process, an annotation tool was created. This tool, in comparison to other programs such as Prodigy or Label Studio, allows the user to perform all actions by keyboard button presses instead of using the mouse. This can greatly speed up the process, since doing everything with a mouse becomes tedious quickly. Additionally, the tool can be adjusted to the demands of the user and Stater. This allows for the pre-annotation of entities by applying advanced regular expressions to the input text, such as e-mails, websites and attachments. Another option is to skip some sentences entirely if they were annotated previously or if there are no entities within a sentence. Furthermore, it is in the possession of Stater and does not require extra costs that would have been inferred if external tools were employed. However, creating a different annotation tool also comes with a few drawbacks. For instance, to accomplish these functions, the program should be able to split sentences and tokens, since looping over all the letters in the document would be impractical. This can allow for some illogical positions or annotations to occur in the training and test set. However, by using the SpaCy token and sentence splitter and adjusting them for the annotation purposes, these drawbacks are mostly negated and should not impact the capabilities of the NER model. Another drawback is that this tool may need to be maintained in some form or another if the company wants to use this tool for other purposes in the future.

To make it possible for SpaCy to split the sentences in the document, it is assumed that the text between any line breaks in each of the datasets represents a body of text that is separate from the other bodies of text in the document. As a result, if a body of text, such as a paragraph, is split by a line break, they will be considered separate bodies of text and will thus be annotated separately. This makes it much easier and faster for the program to properly split sentences. Additionally, any irrelevant sequential whitespace characters or symbols will be either removed entirely or replaced by a single whitespace, as they will not contribute anything to the learning of the NER models. Next, any empty lines, due to such preprocessing, will be removed, since this implies there is nothing to annotate nor to learn. After finishing this initial pre-processing, slightly different pre-processing methods are used for each of the datasets respectively. This is done to accommodate for the distinct formatting of the e-mails specifically when compared to the text inside the inquiry dataset. This can also be seen as a drawback of the tool, since specific pre-processing methods may be required to handle different documents properly if they need to be included in the annotation process. The only addition for the sentence-splitter of the inquiry dataset is the replacement of multiple hyphens by one whitespace, since such patterns interfered too frequently with the tokenisation process. It should be noted that all of these steps can influence the results negatively, since the texts themselves are changed a bit although it is expected that this influence is negligable.

4.3.1 E-mail Preprocessing

After splitting the sentences and paragraphs in a document using the line breaks, these bodies of text need to be split further, since they may contain more than 1 sentence. Since the NER models learn the task per sentence, it is crucial such sentences get split properly. This can be done by utilising the Dutch SpaCy sentence splitter on all lines within the e-mails. However, this can create undesirable formatting in a lot of cases if performed on all sentences. For instance, using this splitting process on e-mail headers or on zipcodes will result in the split of such headers or entities into multiple sentences and thus creates undesirable sentence splits. This is mostly due to the SpaCy model not being accommodated to the format of e-mails within the financial domain. To improve this process, the e-mails will be processed according to the steps listed in detail below. After using these exact steps, the document is ready for the annotation process. The pseudocode in 2 depicts all the steps discussed in code form. The last step of this list is also used for the inquiry dataset but without the previous steps, since there is no special formatting necessary for these documents.

1. E-mail headers (“Aan:”, “Van:”, etc.) will not be split on sentences further, since headers equal, in principle, to only 1 sentence. Therefore, a regular expression is used to detect such instances, which is equal to the following pattern: “`\w{1,15}:`”. This regular expression tries to match the beginning of the sentence with a word that has a maximum length of 15 and is followed by a colon directly afterwards. It is unknown whether various headers are included in each document, such as attachments or extra senders. Therefore, no further assumptions can be made on the size, number or length of these headers. Despite this fact, these splits happen to work wonderfully and account for nearly all header-based situations.
2. After processing these headers, it will be checked whether the mail is a part of the Sent Items box by looking for an identification header (header containing information about the document, including task numbers). If there is no such header, then the algorithm will move to step 4. If there is such an identification header, no sentence splitting with the Dutch SpaCy model will take place inside this header and the algorithm will move to the next step.
3. If the e-mail indeed contains an identification header, then the identification header will be parsed and a follow-up e-mail header check is performed. This is done,since everything after this second header is part of the original e-mail, which includes the header of this e-mail.
4. Next, the same e-mail header check is performed again, since each e-mail text can contain multiple responses to the original e-mail. If there are indeed no headers currently, the program moves to the next step.
5. Since there are no e-mail headers detected, it is checked if the current body of text needs to be split, since it can contain more than one sentence. This is done by first applying two separate checks, namely a regular expression and a length check. The regular expression is equal to “`\.[A-Z][a-z]+(?=[\s])`”, which implies that the program tries to search for a pattern, where the string starts with a dot, followed by a white space, a capital letter, one or more small letters and a whitespace character. In multiple languages, including Dutch and English, such a pattern indicates a separation of two sentences. The other check examines the length of the current line of text, since short sentences are not worth considering for the SpaCy sentence splitter. Moreover, many other errors are avoided with such a check and speeds up the splitting process. Currently, this length is set to 20 characters, which should be sufficiently large to filter out the smaller sentences. If both of these checks hold, then the Dutch SpaCy sentence splitter will check whether to split the line into multiple sentences and performs this split this is deemed necessary.

Algorithm 2 : Processing Sentence Splitter E-mails

```
for Every Document do
    if Document contains useful information then
        Check for initial e-mail header
    end if
    if Document contains indexation then
        Check for this indexation and extra headers
    end if
    for Each sentence do
        if E-mail pattern is detected then
            Start e-mail-header processing again
        else if Sentence pattern is detected and current sentence > 20 characters then
            Use the Spacy sentence splitter with the large pretrained NL model
        else
            Continue to next sentence
        end if
    end for
    Return a list of sentences for this document
end for
```

4.3.2 Annotation Tool

To better understand the reasons for creating a tool separately from existing applications, it is important to first discuss the tool that was made available by the SpaCy creators. This tool moves through each sentence in a document using the sentence splitter of the corresponding model (Dutch in this case) and pre-annotates the named entities in the text according to their default NER schema. Although this may sound useful for this project at first, it should be noted that, if this model does not annotate or process the sentence correctly, these marked entities will have to be corrected each time. Based on some initial experimentation with the

provided Dutch models, these expectations were not met. This is later confirmed by the experiments that were performed. For reference, if the default large Dutch SpaCy model was used on the already annotated test set, without training the models for this dataset, it would achieve a F1 score of approximately 10 - 15%. Therefore, approximately 85% of the data was marked incorrectly or was not detected at all. Moreover, new labels were also added and thus the old schema does not help as much with pre-annotating sentences using the default model. As a result, it was decided to recreate this tool and improve upon its usage to increase the speed and flexibility of the annotation process.

The new tool works similarly, since it moves through a document and will try to annotate this document on a sentence level. However, before moving on to the sentences in each document, the tool will give an overview of the document to be reviewed. The annotator can then choose whether to annotate this file or not. For instance, if an individual file contains little useful data or the file contained a relatively large number of errors due to the PDF-to-text conversion, it can simply be skipped and the annotator can move on to the next file. If the annotator decides to annotate the file, they will be presented with a few options. For each sentence in the document and for each token, the annotator can decide to mark a token as an entity or not. If there are no relevant entities in the sentence, the annotator can move to the next sentence. When a mistake is made, the user can go back to either the previous sentence or restart the current sentence. Moving back multiple sentences can result in nested issues and hence the user can only go back one sentence at a time. In addition to avoiding such nested problems, it encourages the annotator to mark sentences carefully and decisively. If the token in question is an entity, the user can mark it as one of the pre-defined labels and a spelling check is performed to check whether the label was correctly written. If this label is correctly spelled, the program moves either towards the next token or sentence, depending on its position in the current sentence. If there is a next token and this token is not an entity or not the same entity, the span ends automatically at the previous token. In contrast, if the next token is part of the current token, then this entity will be automatically included in the previous span. The annotator can also decide to end the current entity span so that the next token becomes its separate entity. This is useful if two e-mail addresses occur next to each another for instance. The user can also decide to skip a sentence entirely and not include it in the SpaCy train/test set at all. After moving through the entirety of the document, the results are saved automatically and the annotator will move to the next document. An overview of all the actions provided by the tool can be found in List 12 in the Appendix.

4.3.3 Rule-/Pattern-matching system

As was mentioned previously, one of the advantages of creating a system manually is that it can be adjusted to the wishes of both the user and the company. To further speed up the annotation process of the training data, sentences are annotated beforehand for the user and works similarly to Prodigy. However, in contrast to Prodigy, this is done using regular expressions for only specific entities. Entities detected through these means are *PER*, *GPE*, *POST*, *MONEY*, *ID*, *EMAIL*, *DATE*, *TIME* and *FILE*. The system of regular expressions is directly related to the annotation scheme shown in Table 2. These patterns were either crafted manually or were based on the paper by Menger et al. (2018), which successfully implemented rules and patterns for the medical domain. Since some of these entities are considered general, such as postcodes or dates, these patterns and ideas can be adapted to the financial services domain. Additionally, some of the manually created patterns were based on the initial efforts of this paper. Since these patterns can create huge biases in the results, this system is only used for the training data, such that the test data is not influenced by these biases. Although this system is not perfect, it can alleviate a lot of annotation work. Moreover, if one of these patterns turns out to not be correct for a specific case, the user can simply restart the whole sentence and annotate everything manually with the press of one button, which rarely slows down the annotation process.

The full system of regular expressions can be found in Table 13 in the Appendix. The first few patterns are mostly used as a basis for other patterns, since, generally speaking, only combinations of patterns form an entity. These patterns are then used throughout multiple other patterns. In addition, some general lexicons and datasets are used within these regular expressions to find a large number of specific entities and thus enrich the system. For instance, a list of known Dutch cities, towns and provinces are used to automatically detect entities of the *GPE* label in text. Furthermore, multiple databases from Stater are used to automatically retrieve various columns, such as person surnames and identification numbers. To complement the surnames, an open-source dataset of first names is retrieved and used in the name patterns. Although most of the patterns are covered in this table, some are adjusted later when creating the models and these changes will be thus explained later in the thesis. Moreover, some of these labels, such as the *PER* label, will not be used for the models, since these patterns and databases only provide the basic knowledge of such labels. If these systems ought to generalise to new entities, then the models will need to learn them without the use of such regular expression systems. Additionally, some more entities could have been detected using this process as well, such as *STREET*, but were not implemented due to time restrictions.

Table 3: Post annotation statistics on the files, sentences and labels annotated.

Category	**E-mail**		**Inquiries**		**Totals**				
Labels	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**E-mails**	**Inquiries**	**Total**
Files checked	406	210	98	55	504	265	616	153	769
Files annotated	378	111	60	32	438	143	489	92	581
All sentences annotated	13919	4426	6367	4023	20286	8449	18345	10390	28735
Unique sentences annotated	6969	2551	4322	2570	11291	5121	9520	6892	16412
All annotated labels	15424	4901	7065	3930	22489	8831	20325	10995	31320
Unique annotated labels	9375	3394	5802	3041	15177	6435	12769	8843	21612
Difference sentences in %	50%	58%	68%	64%	56%	61%	52%	66%	57%
Difference labels in %	61%	69%	82%	77%	67%	73%	63%	80%	69%

4.4 Post Annotation Statistics

The annotation process itself took approximately 130 hours to complete and Table 3 shows the most important and general statistics of this process. It should be noted that the statistics from this table originated by randomly choosing from the pre-selected documents discussed in the Data Exploration and Selectionsection. The only exception is the test set for the inquiry dataset, as was discussed in this same section. Therefore, these results may be a bit biased towards either specific email files or specific inquiry files, since this is only a small selection of the total potential files to annotate. However, annotating more files to reduce the bias would require much more time and effort. Given current time constraints, this is not possible and will be left to future work. In addition, it should be noted that the threshold of the cosine-similarity was adjusted midway through annotating the emails, since too many similar files were observed. Although these files were useful, annotating too many of such files would again create large biases. Therefore, this threshold was decreased from 85% to 70% and afterwards only documents from this list were selected.

The table shows that a total of 28,735 sentences were annotated, which contained a total of 31,320 entities. This implies that, on average, every sentence contains at least 1 entity. The total number of files checked equals 769 and the number of files annotated equals 581. However, there are a large number of duplicate sentences within the data. The total number of unique sentences equals 16,412 with a total of 21,612 entities annotated. These statistics show that, for the unique sentences, more entities were detected in comparison to the previous statistics. In addition, these numbers show that approximately 60% of all sentences are unique on average. This does not have to be an issue when training the NER algorithm, although it could influence the generalisability of these models. Therefore, this will need to be tested in the next parts of this research.

Moreover, the total number of e-mails annotated is equal to 489, while the number of inquiry documents annotated equals 92, which results in a total of 10,390 sentences annotated for the inquiry dataset in comparison to the 18,345 sentences annotated for the e-mails. This needs to be taken into consideration when discussing the distribution of the entities across the datasets, since it biases the e-mail dataset in terms of labels. However, from the results displayed in this section, it becomes evident it is also the most balanced dataset. This was the main reason behind annotating more e-mails in comparison to the inquiry dataset. In addition, more errors were noted for the inquiry dataset due to the PDF to text file conversions. Although this works fairly well for text within the PDF files, it did not work as properly for either rotated documents or documents containing certain images. Therefore, some files were skipped from this dataset and overall less files were annotated. From the statistics in Table 3 it can also be deduced that the Inquiry dataset contains more unique sentences in comparison to the e-mail dataset. However, this can be both a result of having less files annotated in the dataset and having annotated more specific directories before continuing with the more similar files as was mentioned during the Data Exploration and Selection section.

The distributions of the annotated entities are shown in Figures 6 and 7 below. The first graph shows the distribution of the labels across the two datasets, while the second graph shows the distribution of the labels

when combining the two datasets. From Figure 6, it is evident there were both similarities and differences when annotating these datasets. For instance, the label *EMAIL* showed up frequently in the e-mail dataset, while these entities appear only rarely in the inquiry dataset. This is a logical consequence of annotating email texts, since emails are bound to occur in such documents. A similar reasoning can be applied for the labels *ORG*, *PER*, *FILE*, *TIME* and *WEB*, which either occurred often in the header or in the ending of the email texts. Considering the relatively less number of sentences annotated for the inquiry dataset, there are relatively more instances of *CARD*, *DATE*, *ID* and *MONEY* labels in this inquiry dataset. The other labels have much more equal distribution among the two different datasets.

When combining these two datasets into one, it can be seen that the labels are relatively well distributed. However, there are two labels in particular that do not occur frequently, namely the labels *FUNC* and *NORP*. This is not entirely unexpected, since one was manually added for the e-mails and the other was added to better conform to the requirements of the departments at Stater. Since these labels do not occur often enough, they are merged with the labels *DEP / ORG-RELATED* and *GPE* respectively as both categories have something in common with these rarer labels. Given the low number of occurrences of these entities, this should not impact the results while still being able to mostly recognise such entities. Another observation to note from both of these graphs is that the labels *DEP*, *EMAIL*, *ORG*, *POST* and *WEB* contain an enormous number of duplicates resulting in a twice as many entities for those specific labels. From Figure 6, it can be surmised that the re-occurrence of entities mostly originates from the e-mail dataset. This is a consequence of most e-mails being sent to and from the company Stater, through which many entities are used numerous times in these e-mails. For instance, the company name "Stater" occurred approximately 738 times throughout the entities in both datasets. More statistics on the distribution of the labels specifically are shown in Tables 14 and 15 in the Appendix.

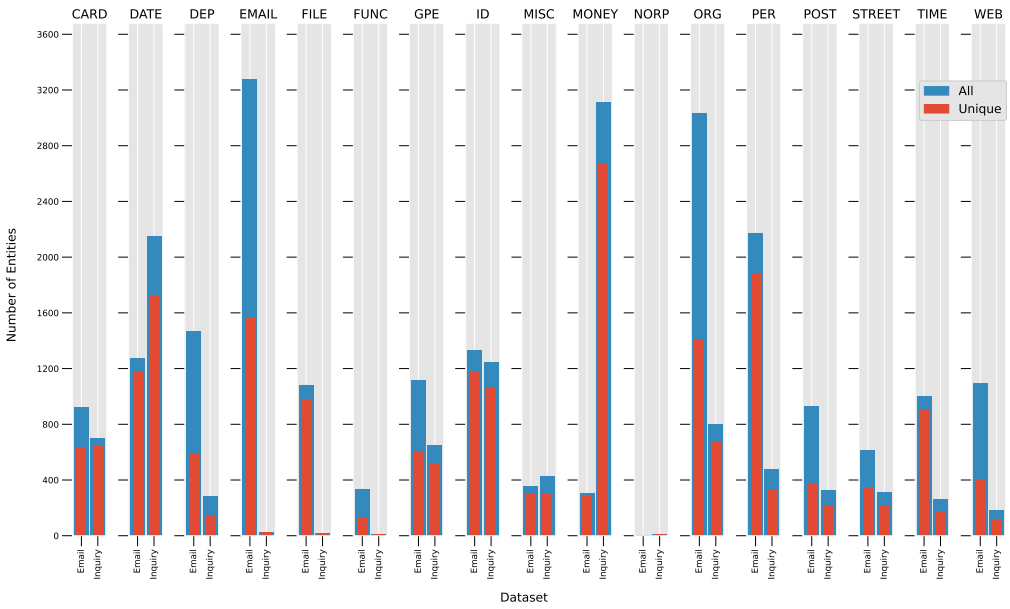

Figure 6: The distribution of the labels across the two datasets. It includes all mentions and all unique mentions of a certain label for a given dataset.

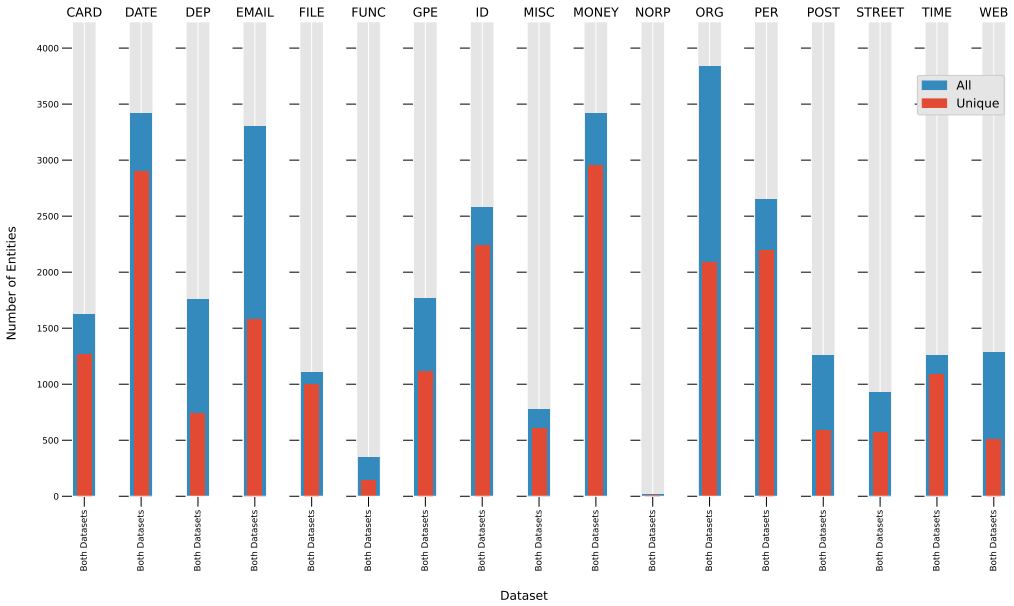

Figure 7: The distribution of the labels when combined into one dataset. It includes all mentions and all unique mentions of a certain label for a given dataset.

5 Model Descriptions

5.1 Introduction

In this section, the three main models for the NER problem will be described. The first approach will elaborate on the open-source software library SpaCy, which has a Dutch NER model available that will be fine-tuned on the annotated datasets. The second model and third model are variants of the BERT model, which has shown great potential for the task of NER in the literature [23]. There are two main reasons for choosing these two types models for the task of NER. First, Spacy is an open-source software that will be updated and improved in the coming years with new models, which makes the maintenance for Stater of this model more straightforward when compared to the BERT model. On the other hand, BERT is the current state-of-the-art model for many NLP related tasks and also showed huge potential for the NER task. Since Spacy has not trained their transformer model on the Dutch language yet, it will be interesting to see the difference between a transformer-based and non-transformer based approach. Second, the manually created model allows the user to create their own features and use them in the model, which is currently not possible with SpaCy and thus makes the BERT approach more flexible to Stater. Third, two variants for the BERT model are considered, since they have distinct training regiments and architectures. All the other models investigated in the Literature Study section will not be considered further in this thesis, since they are unlikely to outperform either of these approaches. Additionally, feature engineering will not be done in this thesis due to time restrictions. All experiments related to all three of these approaches are described in the Set-up Experiments section.

5.2 SpaCy's NER approach

5.2.1 Background

SpaCy is an open-source library for Python that was created by the company Explode AI, which was founded by Matthew Honnibal and Ines Montani. The library has pre-trained models for multiple languages and various pipelines related to language processing, such as parsing, lemmatisation, tokenisation, sentence splitting and more, some of which were used in the Data Exploration section. Explosion AI uses the Thinc library as a basis for their SpaCy applications, which is a library specifically created by Explode AI for simplifying various processes related to machine learning and NLP. There are multiple reasons for choosing this specific open-source software for these experiments. First, SpaCy is one of the most popular open-source libraries for Natural Language Processing in Python and thus is being updated regularly. This is advantageous for Stater, since it implies that the performance of the NER model will continue to improve by introducing and updating their current models. Moreover, SpaCy was used for various other projects within Stater and thus can be

better mainstreamed into the processes of Stater as a result. Other open-source libraries could help improve performance, since they use different methodologies, but this will be left for future work.

5.2.2 Model Description

Since there is little description on the current models used by the SpaCy library, most of the information will originate from a video that was created by Matthew Honnibal for SpaCy version 2 in 2017, where he elaborately explains most of the decisions that were made for the NER model [2]. However, it should be noted that this video was made approximately 5 years ago as of writing this thesis. Therefore, this explanation may be dated in some aspects, although the main idea should be similar.

The main idea of the SpaCy model can be divided into 4 steps:

- The creation of Bloom-embeddings to embed the raw text into number-based representations.
- The encoding of the embeddings using a Trigram CNN layer, which recalculates the meaning of the embeddings by incorporating the context of the words in these embeddings.
- Using an attention layer to include extra features in the model and reformat the encodings into a single vector.
- A prediction layer to determine the probability of each possible action given the output from the attention layer.

Each of these four steps will be elaborated in the paragraphs below.

Bloom Embeddings:

The first step to understanding the embeddings within the SpaCy framework is to view each sentence as a document, which thus can also be seen as a matrix. Each document consists of words, which are equal to the rows in a matrix. The first step is to embed each word in all documents given by the user. Normally, words are represented by encodings that are obtained from a fixed knowledge base. So, if the word is known in the knowledge base, it will get the exact encoding from this source. However, each word that is not known by the knowledge base will get the same out-of-vocabulary vector. For instance, if the words "Stater" and "Department" are not known by this knowledge source, both words will get the same encoding. This is also the case when using embeddings within BERT, since its tokeniser uses a similar solution in the form of the "unknown" embedding. This can have detrimental effects for words that are not known to the knowledge base, since they will be seen as similar and thus such embedding tables will not generalise well to new data.

To combat this issue, SpaCy makes use of the Bloom embeddings, which creates unique vectors for each word it encounters. An example of this process is shown below in Figure 8. Such embeddings are created by first extracting four features for each word, namely "norm", "prefix", "suffix" and "shape". "Norm" (sometimes referred to as "Orth") refers to the normalised form of a word, which can be obtained by replacing all upper-case characters by lower-case characters, lemmatising the given words/tokens, etc. "Prefix" and "Suffix" try to split the word into logical parts, such as "Apple" being split as "A" and "ple" as is shown in Figure 8. "Shape" refers to the syntax of the word. For instance, the word can contain one upper-case character and multiple lower-case characters, contain upper-case characters only, contain both numbers and letters, etc. The "Shape" feature is especially useful when encountering new words, since their shapes may be similar to the words that are already known and thus could help the generalisability of the model. The features used for the embeddings can be categorised as the columns in the document matrix. These features were likely chosen, since affixal features in particular were shown to perform well for the task of NER [90].

After extracting each of these features, each column is encoded using a technique called the "Hashing trick", which encodes each feature x times using a different encoding each time and using a specified embedding table. These encodings are also referred to as "keys" by Explosion AI. Normally, hashing each feature will result in many words obtaining the same key. However, this is undesirable, since many different words would then get the exact same key, which can consequently lead to confusion for the model. Therefore, each feature is encoded a total of 4 times in the SpaCy model as to obtain an unique encoding for each word. Consequently, if a new word is detected, it will also obtain a unique encoding when compared to many other techniques, making it easier for the model to learn new words. However, this also implies that the model becomes much more difficult to train, since every word and every variation of these words gets an unique encoding. This is a problem that can mostly be remedied by simply pre-training the model once and then fine-tuning it for domain-specific tasks. Moreover, NER models are also quite likely to overfit and thus this makes for an overall reasonable trade-off.

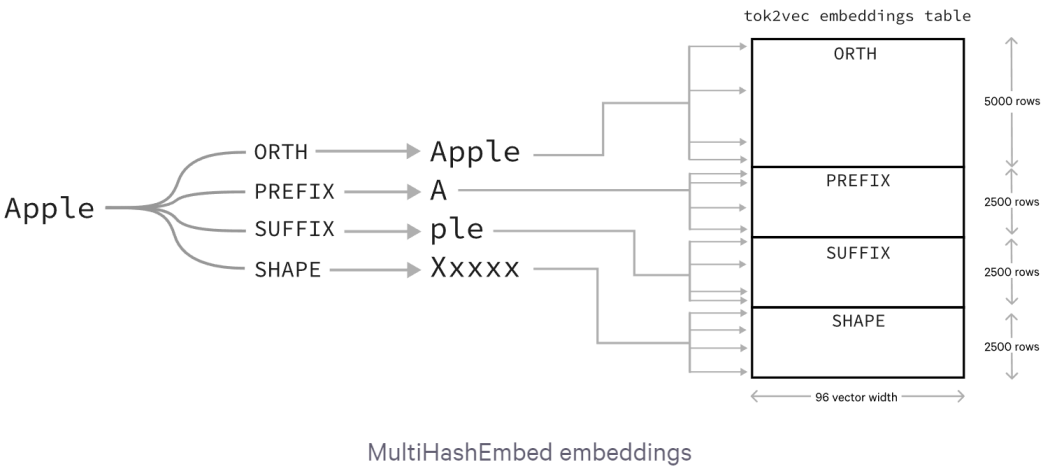

Figure 8: An example of how the Bloom Embedding roughly work for the word "Apple". Here, "ORTH" represents the "Norm" of the given word. This image was created by Honnibal et al. (2022).

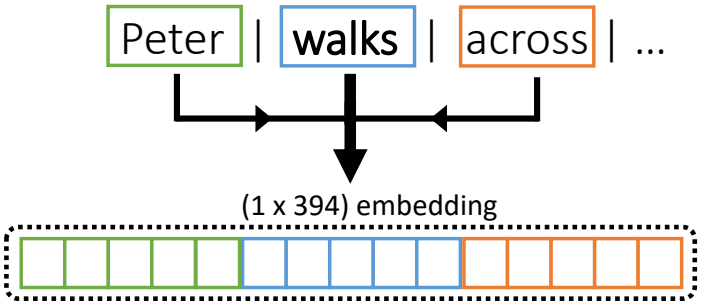

Figure 9: A figure showing how each word is embedded using the context of the sentence. In this visualisation, the word "walks" is embedded accordingly.

Trigram CNN:

The next step is to use layer normalisation on the previously embedded input and insert them into a CNN layer. Normally in computer vision, CNNs are used mainly for dimensionality reduction of images. However, the layer used within the SpaCy architecture is the so-called Tri-gram CNN layer (more generally known as a n-gram CNN), which is primarily used to look for a window of words outside of the current word so that the context of those words is taken into consideration. The range of this window is 1, which implies that only the next word and previous word are considered for each word in the sentence. This creates a vector of length 394 by concatenating the two neighbouring words to the current word and include redundancy on either side of the window. Afterwards, the entire sentence of embeddings is put into a Multi-Layer Perceptron (MLP), which contains a total of 4 layers and gives an encoding back of length 128. This creates an effect, where each word in the sentence uses their neighbouring words to create a new meaning for the current word. Moreover, through this process, a decay effect occurs the further the inputs move through the network. This is due to the context range increasing after each layer and thus including more words. However, these words are also encoded multiple times, which makes their encoding effect on the current word much less than in the first layers.

To better understand this encoding process. two visualisations have been provided and are found in Figures 9 and 10. For these visualisations, consider the following sentence: "Peter walks across the street towards the supermarket Albert Heijn." Assume that each word in this sentence gets its own vector of length 394 by considering the context of each word. For instance, the context of the word "walks" is in this case "Peter" and "across", as is shown in Figure 9. Afterwards, the 10 * 394 matrix is used as input for the Multilayer Perceptron, where *10* is the number of words in the sentence. In the first layer, the context gets embedded into

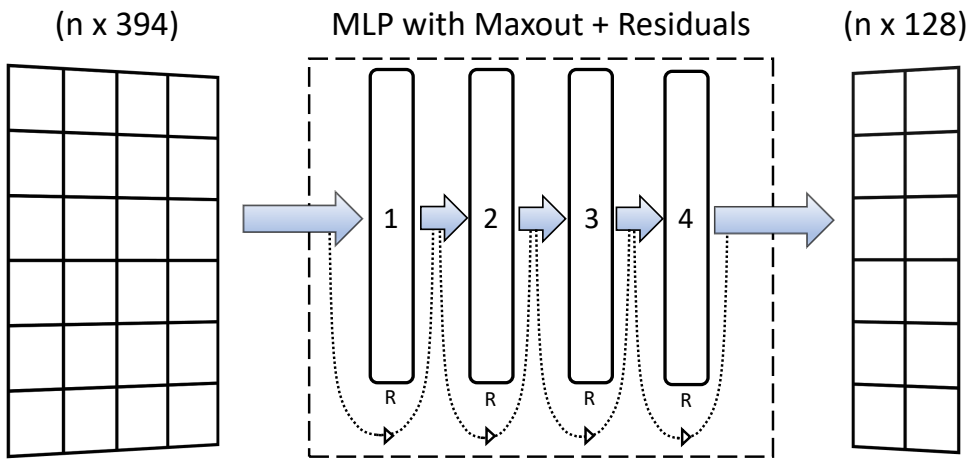

Figure 10: The architecture of the model after embedding each word. *R* represents the residual layers in the MLP. Additionally, Maxout is used as activation function after each layer, since, according to Hannibal, this obtained slightly better results [2].

the current word to create a new meaning. However, for each following iterations, this context range increases creating new meanings at each layer of the network. For instance, if the current word is "walks", then "the" will be used to enhance the meaning of "walks" and vice versa.. On top of that, since this is the second layer, the meaning of the word "the" already includes the context of its direct neighbours "across" and "street". In the fourth iteration, the scope of the current word is then equal to 4. However, since the meaning is adjusted multiple times throughout this network, the effect of the context is marginally smaller. So, when considering the word "walks", the word "towards" will not adjust the meaning of "walks" as much as the word "across". This is the decay effect that was mentioned previously. The creators do not see much motivation to increase the number of layers, since this is already a satisfactory window of context to consider [2].

There are multiple reasons for choosing to use a CNN layer here instead of, for instance, a BI-LSTM layer. One of the major reasons is that a CNN layer is much easier to train compared to a BI-LSTM layer, since the CNN layer can be parallelised. Moreover, inside the CNNs, you can use Residual layers, which sum the input and initial output of each layer and feed this into the next layer, as is shown in 10. This gives the model incentive and bias to keep things the way they were without swapping the contexts around too much. A last reason is that the creators found that it makes more sense to simply cut off parts of the documents in order to get context rather than inserting the entire document into the model, as is done with a BI-LSTM layer.

"Attention layer" and Transition-based approach:

After using the CNNs, the resulting embeddings are then put into an "attention layer". Quotation symbols are put around this wording, since the creator indicated that it is not exactly attention, but that he tries to imitate its workings. The idea behind this layer is that the matrix of words in a document is transformed into a vector by further considering the context in each word. This is achieved by using the Query, Key, Value framework from the attention layer. However, instead of using the weighted sum in the layer, the model creates extra features for each word and incorporates the neighbouring context of these features for neighbouring words. This is different from a CRF model, since this model bounds the number of previous decision. Moreover, instead of using Viterbi encoding, greedy processing is used to enable feature-based functions. The workings of a true attention layer and Viterbi encoding will be described in the BERT-based approach, since it is much more appropriate there.

The last layer in the model is simply the prediction layer, for which the creators used another Multilayer Perceptron. In contrast to the BERT output layer, where the IOB-scheme is used, SpaCy adopts the IOBES scheme and allows for a more complex way of labelling entities [105]. In other words, it allows the model to output specific actions for entire spans of words. As a result, SpaCy is able to use a transition-based approach for solving the task of labelling entities. In short, this approach sequentially moves through a document, where either the model adds a label to the current word or it changes the internal state of the system given the current

position of this system in the document. For instance, the system can add a new word to a stack, it can give the stack a label or it will not label anything and continue to the next word. There are several advantages to using this system, which includes better excluding of invalid sequences and defining arbitrary features. Using this transition-based approach, the network will decide the action for each step in the document based on the maximum probability calculated with the output of the neural network. Afterwards, SpaCy can label the sentences accordingly and return the desired result to the user.

5.3 BERT approach

Before discussing the BERT approach, it is informative to first elaborate on the details of the Transformer model, since a major part of their architecture is used multiple times inside the BERT architecture. Thus, by understanding this architecture, the predictions and errors made by the models can be better understood. Next, the architectures and details of the BERT models will be discussed, which includes the transfer learning of this model to other tasks such as NER and the difference between the BERTje model by de Vries et al. (2019) and RobBERT model by Delobelle et al. (2020). Afterwards, the CRF model will be explained with the Viterbi encoding, which is used on top of the BERT model to hopefully help the model create better entity boundaries.

5.3.1 Transformers

Introduction

To make the difference clear between prior models and the new transformer model, the history of sequence-based models will be briefly discussed first. As was explained in the Literature section, before the transformer model was introduced, the Natural Language Processing research community often used the Recurrent Neural Network (RNN) as a basis for text analysis. More specifically, the Bidirectional Long-Short-term Memory (BI-LSTM) model was often used to obtain state-of-the-art performance in many NLP-related tasks such as NER [107, 108, 109]. This is a consequence of this model being able to learn on a sequential-basis. This implies that the input is read from left to right as is the norm for languages such as English or Dutch. Therefore, these models were able to learn and understand text more naturally compared to other models at the time, such as the feed-forward neural networks. The idea of attention was already applied in the more complex versions of the LSTM networks, since it is able to deal with the vanishing gradient problem rather well. Moreover, it was also used in different models and tasks, as is shown by Jaegle et al. (2021). However, it was never fully adopted into its own model until the paper by Vaswani et al. (2017) was released in 2017, which gained a lot of traction in the research community and lead to the creation of the Transformer model and consequently BERT.

In the paper by Vaswani et al. (2017), the attention mechanism was used to mimic the concept of cognitive attention by making a distinction between important and non-important parts of the input. In this way, only the parts crucial for understanding text are considered while the rest is left out of the model, as would be done with memory-units in the LSTM networks. The main difference between transformers and the former RNN models is that transformers consider the entire sentence at once when evaluating the input, while the RNN had to sequentially move through the sentence. This implies that the interaction is checked between each individual input and all other inputs. This omni-directional approach has two advantages over the RNN. First, it considers all words in a sentence equally at first and thus each words is taken into account without losing meaning when the distance between words becomes larger. Second, this technique is parallelisable, since these are matrix-based operations, making the training process much faster. This in turn has many other advantages, such as being able to learn more data and thus increase the model's capabilities in understanding text.

The mathematics

Since the Transformer model is critical for understanding the BERT model, this thesis will also dive a bit deeper into the mathematics used within the Transformer model. Transformers consist of 2 main components, namely the encoder and decoder stack. Each of these stacks contain multiple encoders and decoders, which do not have to be identical in terms of parameters. The encoder itself consists of 2 components, namely the self-attention layer and a neural network. The neural network contains a few linear layers that use the Rectified Linear Unit (ReLU) as their activation function. The self-attention layer is more complex and its mathematics are discussed in the next paragraphs. Figure 11 shows the general architecture of a single transformer.

The self-attention layer starts with three sets of scaled dot products, which will be referred to as "Key", "Query" and "Value" and were named after their utilisation in retrieval systems [3]. These matrices are obtained through linear transformations from the input using weight matrices. These weight matrices will be updated during training to improve the model's encoding capabilities. After the creation of the Key, Query and Value matrices, these matrices will be multiplied with each individual input. Attention scores can then be calculated by taking the dot product between each individual Query and all Key matrices. Next, the weights are normalised by applying the Softmax function to each of these attention scores. Afterwards, these attention scores are multiplied with each of the Value matrices to obtain the "weighted values", which are then summed according to their

weight for one attention score per input. Therefore, the input dimension of this self-attention layer is equal to its output dimension. This process is visualised in Figure 12. The parallelised version is called Multi-head attention, which splits the Key, Query and Value into n different splits, which are combined together at the end to produce the final score again. This also means that the weight matrices are split into multiple matrices in comparison to the original three matrices.

The next step is to combine both the input and output of the self-attention layer, which is done through so-called residual connections. Such connections can help retain the original form and meaning of the input while also adjusting them accordingly to the output provided by the attention layer [2]. Additionally, layer normalisation is used to stabilise and speed up the learning process and thus avoid vanishing or exploding gradients. After layer normalisation, these values are used as input into a neural network containing a few linear layers. The output of the neural network is again combined with its corresponding inputs through residual connections to finalise the architecture of one encoder. However, the power of Transformers comes from the concept of using multiple encoders sequentially in the model, through which different information can be obtained for each encoder used. After moving through the stack of encoders, the input will move through a stack of decoders. These decoders are mostly similar to that of the encoder, although there are 2 important differences. First, the encoder layer is switched to a decoder layer, where the interaction between each target word is checked, whereas this was the input word in the encoder layer. Second, it makes use of an extra encoder-decoder layer, which computes the interaction between each input word and each target word. This layer is put in the middle the self-attention layer and neural network, while also making use of layer normalisation and residual connections, as is shown in Figure 11.

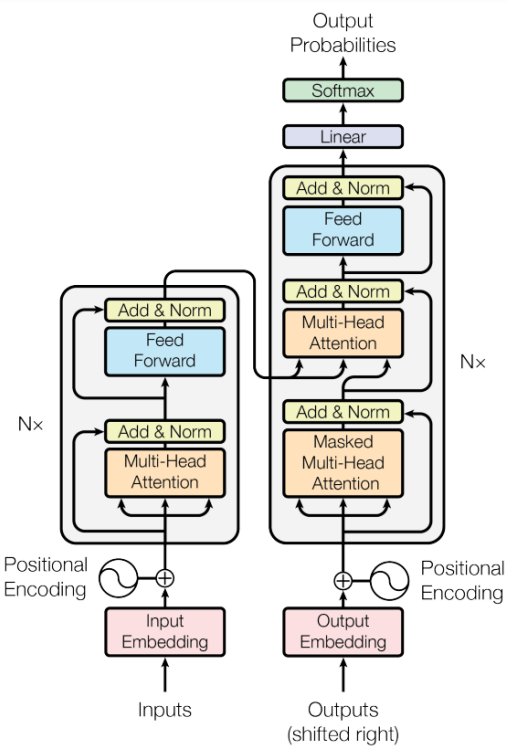

Figure 11: The architecture of the Transformer model as defined by Vaswani et al. (2017).

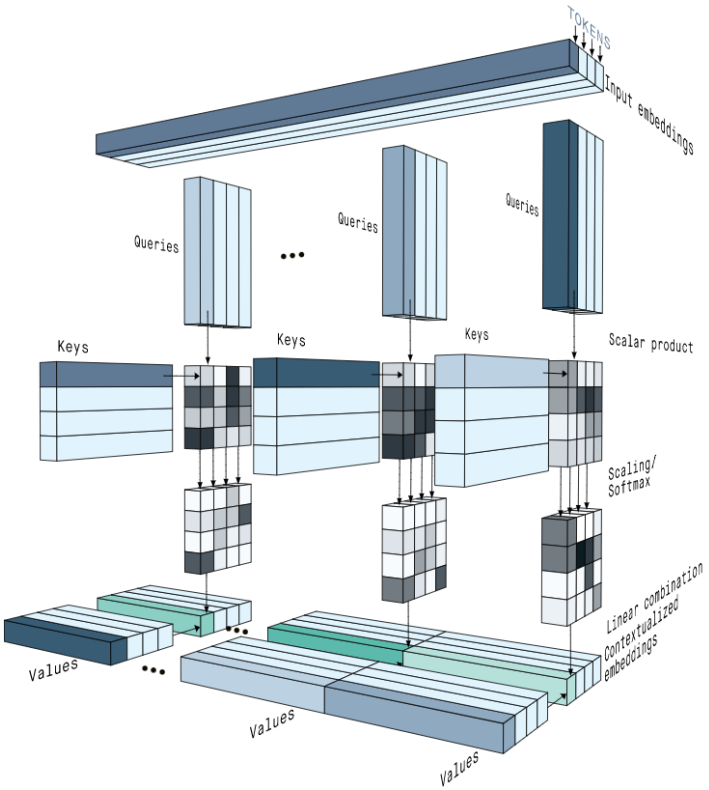

Figure 12: The architecture of the multi-head self-attention layer. This image was created by Futrzynski (2020).

5.3.2 BERTje and RobBERT

BERT is the abbreviation for Bidirectional Encoder Representations from Transformers and was first introduced by Google in 2018 [6] and has gained a lot of attention due to its excellent performance in NLP-based tasks. The English version of BERT was trained on data that originates from the BooksCorpus [111] and English Wikipedia. The main concept behind this model is to pre-train unlabelled text embeddings from both directions, hence the bi-directionality in the model name. However, this is only partially true, since it also manages to include context from all directions, due to the nature of transformers. As was mentioned previously in the transformer section, each input is connected to all other outputs. Therefore, this thesis will refer to this phenomenon as omni-directional or non-directional rather than bi-directional. Through this non-directional learning, it is able to adapt to much more complex and variant texts, which makes it a good fit for language-based tasks. Another important aspect of BERT is that transfer learning and fine-tuning are critical for BERT's popularity. Although BERT performs especially well for tasks such as Next Sentence Prediction (NSP) or Masked Language Modelling (MLM), which were the tasks it was originally trained in, this task can be changed by adjusting the last layer of the BERT model. The adjustment of this last layer is also known as transfer learning. This is possible for BERT, since the understanding of language mostly originates from the transformers within the model. This is in correspondence to the idea that NER considers two tasks, namely the embedding of the documents and the classification of the labels. Moreover, if the model needs to be used for a specific domain, it can be fine-tuned to fit this domain and thus update all weights in the model. To fine-tune this for the task of Named Entity Recognition, a classification layer will be used as the last layer of the model. With this classification layer, the model will be able to check which token belongs to which label. In order to achieve this result, the token-based version of BERT will be used throughout this project.

BERT mainly consists of a large number of Transformers connected to one another, although the key difference is that BERT only uses encoding layers. This is the case, since BERT's original goal is to generate a language model, for which only the encoder part is necessary. Due to the complexity of language and its use for various tasks, two versions of BERT were created, namely "BERT base" and "BERT large". The specifications of both models are shown in the Appendix in Table 16. The BERT base model consists of twelve encoder transformers that are connected sequentially to one another, as is shown in Figure 13. Through the usage of these twelve Transformers, different aspects of language are captured by each individual transformer within the model. Additionally, GELU and normalisation is used in the classification layer to obtain the final output. Both BERT base and BERT large are discussed here, since both are used in the different models that will be tested throughout

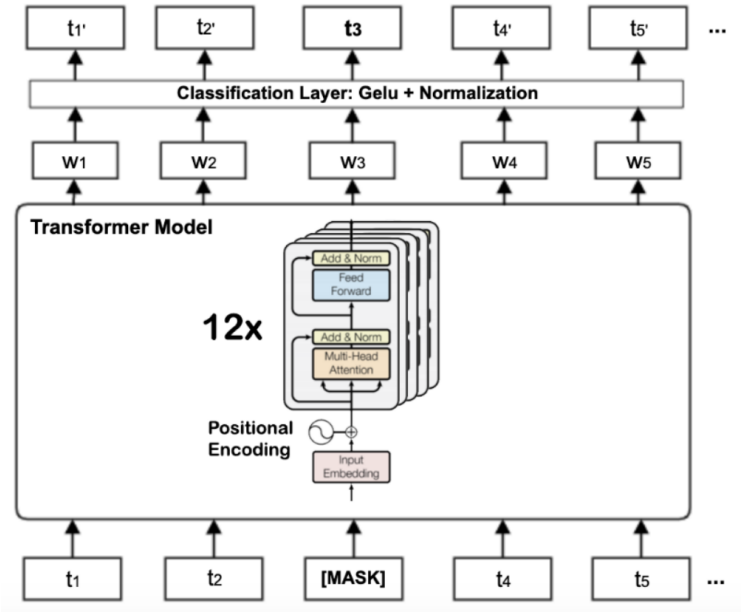

Figure 13: The training procedure of BERT for Masked Language Modelling. This image was created by Khalid et al. (2021).

this thesis. Only monolingual models will be tested, since they seem to outperform the multi-lingual models for monolingual tasks [103, 112]. After some short experimentation within this thesis, this indeed seems to be the case. Both BERTje and RobBERT are considered for this NER-based task, since they are trained on different datasets. Additionally, the architectures used between these two approaches differs, since BERTje mostly focuses on BERT base, while RobBERT used the BERT large architecture in combination with more robust and dynamic learning of tasks [106]. Therefore, it will be interesting to compare both approaches for the task of anonymisation with and without the Regular Expression and CRF-based variations.

In the paper by de Vries et al. (2019) it is shown that, for the task of NER using the CoNLL-2002 evaluation script, the accuracy increased slightly in comparison to the other models. In contrast to BERT base, this model was trained on five different sources, namely Dutch books, TwNC [113], SoNaR-500 [114], Dutch web news and Wikipedia articles. Although most of these sets were used, some data was left out due to quality considerations. As for the paper by Delobelle et al. (2020), the RoBERTa model by Liu et al. (2019) was pre-trained for the Dutch language. The OSCAR corpus [116] was used for training, since this is a large multi-lingual corpus and is larger than the other datasets that were used to train the other Dutch models, such as BERTje and BERT-NL [106]. Another difference between BERTje and RobBERT is that the RobBERT tokenizer recognises symbols, such as the "€", "%" and "@'' symbols, out of the box while BERTje does not include such symbols in their vocabulary. Moreover, the difference in tokenisation can help the RobBERT model outperform the BERTje model, since it may be easier for RobBERT to detect entities such as money amounts and email addresses.

5.3.3 Conditional Random Field (CRF)

For the current architecture, a CRF layer is added on top of the BERT models as a variation of the general BERT model. Moroever, the usage of such a layer has seen promising results for other NER papers [117]. This was mostly done to include a sequential component into the logits that are outputted by the linear layer from BERTje. Logits are defined as the values that are normally put in a softmax layer to retrieve the most likely label or action. Normally, such logits do not contain any sequential component so it may have trouble with identifying the boundaries for long entities. For instance, if there is an entity "New York Times", then the "Times" is often missed or classified separately due to "New York" being seen as a city first while not even considering the next words. One of the better solutions to this problem seems to be the Conditional Random Field (CRF) model, which is used on BERT's logits. Based on the literature studies, it seems that this model has been very effective for this purpose, as more than half of the best models in the paper by Li et al. (2020) used the CRF model as a tag decoder.

The mathematical definition for the Conditional Random Field is shown below in Equation 1, as was defined by Lafferty et al. (2001), and will be elaborated in this paragraph. Given a fixed graph $G = (V, E)$ and given random variables *X* and *Y* over the data sequences and label sequences respectively, then (X,Y) is a Conditional

Random Field, when each random variable Y_v, conditioned on *X*, obeys the Markov property with respect to the undirected graph. In other words, given that *Y* is indexed by the vertices of *G*, the probability of a label Y_v, given the other labels Y_w $w \neq v$, should be equal to the probability of this same label Y_v, given the other labels and the data sequences, such that *w* and *v* are neighbours in the graph *G*j.

The CRF is based on the concept of the Markov Random Field. A Markov Random Field represents an undirected graph between random variables and gives insight into the dependence and independence between these variables. The CRF is a special case of this model, since the Markov property is checked for all nodes *Y* given that *X* is a fixed random variable. Moreover, the CRF is a discriminative model, which implies that decision boundaries are chosen by checking the probability of a certain label assignment. For instance, what is the probability that the label assignment of the word span "Peter likes bread" is equal to "['PER', 'O', 'O']"? With this in mind, the goal of the CRF model is to maximise the likelihood of a certain label combination given the words in a sentence or span. This likelihood is optimised by training the weights in the model. Given that, as previously discussed, many models, including the BI-LSTM model, have issues with deciding these boundaries, it becomes evident that why the CRF is so popular and effective. Hence, it will be used in this thesis on top of the BERT model described previously.

Another way to view the CRF model is that it's goal is to find the optimal path through the undirected graph of labels. This means that the optimal route needs to be determined through all tokens in a sentence. Each token has then the choice between *x* labels, which will be referred to as a layer. In order to find the optimal path through all these layers, the Viterbi algorithm is used in many implementations of the CRF. Viterbi is a dynamic programming algorithm that is used for calculating the maximum a posteriori probability estimate of the most likely sequence of hidden states. The mathematical definition of the Viterbi algorithm can be found in Equation 2. In other words, the goal of the Viterbi algorithm is to find the maximum probability *P* of all possible sequences *S* given observed outputs y_t with final state *k* and the state probabilities $a_{x,k}$ given state sequences x_t. This algorithm calculates the maximum probability for each layer. Moreover, each previous layer is used in the next layer ($V_{t-1,x}$), which makes this a dynamic programming algorithm. Given an optimal candidate for each layer, the algorithms performs a backward pass through all layers and checks whether this is truly the optimal path or if there is a more optimal option. These calculations also use the Markov property, which means that the entire history is known and that this is all that is needed to give the probability of the next label. Not only is this used for the CRF, but also for Hidden Markov models and has seen usage in various fields, such as speech recognition/synthesis and bioinformatics.

$$P(Y_v|X, \{Y_w : w \neq v\}) = P(Y_v|X, \{Y_w : w \sim v\}) \tag{1}$$

$$V_{t,k} = max_{x \in S} \left(P(y_t|k) * a_{x,k} * V_{t-1,x}\right) \tag{2}$$

6 Set-up Experiments

In order to check the performance of the models discussed in the previous section, some experiments need to be set up. First, some small pre-processing steps need to be executed before using the annotated data in the models. Furthermore, there are various implementations possible for the given models and their parameters. Therefore, these implementations and the hyper-parameter tuning will also be described in this section as well. Afterwards, the specific experiments for testing the performance of all approaches will be described. For these experiments, an important component are the Regular Expressions, since it is likely some of these labels can be better solved by simply using complex regular expressions rather than using NER models. For instance, e-mails and website links can generally be recognised by using the "@" symbol and "www." pattern, which enables the use of complex regular expressions and seem to perform consistently well baring some exceptions. To reach this state however, some adjustments need to be made to the existing Regular Expressions shown in Table 13 in the Appendix. Therefore, these changes will be discussed next. Lastly, the evaluation metrics used for these experiments will be elaborated as well, since some new scores and metrics are introduced for these experiments .

6.1 Final Preprocessing

Before moving on to these final preprocessing steps, note that all following steps are carried out for all models, so that the same training set is used for all experiments. The first step after annotating the data is to merge the train and test set of both the email and inquiry datasets into a general train and test set. The train dataset will be split during the training process to obtain the validation scores, which also uses a 1 to 5 ratio. The second step is to transform the document-based format into a sentence-based format. The document-based format was used to preserve the majority of information of each sentence in the dataset. Afterwards, as was discussed in the Annotation section, the labels *NORP* and *FUNC* are merged with the labels *GPE* and *DEP/ORG-RELATED* respectively. Next, if applicable, any entities that will be found with the regular expressions are removed from

each sentence during training. Which entities this entails will be discussed later in this section. This is only done for the data that will be used as input for the models, since these regular expressions should still be evaluated after each epoch. Therefore, to obtain the correct training and validation scores, the models will first predict the labels without the regular expressions and these predictions will then be combined with the predictions from the regular expression-based system to obtain the final scores.

Although the majority of the annotated data can now be directly used for fine-tuning the models after performing these steps, there are some sentences which resulted in errors for the SpaCy model. For instance, the start and end position of some entity labels did not match, where the start position is equal or larger than that of the end position, which should not be possible. Although the origins of these errors could not be determined, the entire sentences with these entities were deleted from the train and test dataset. This was done, since it is apparent that this is an error that can span multiple entities. Moreover, this did not occur often and thus the safest option is to delete these sentences. Additionally, another token span error was detected. This is an error where the model only accepts labels that correspond to the token spans provided by the SpaCy tokeniser. For instance, "Amsterdam" could not be a label in the sentence "... Amsterdam| ...", since the pipe symbol was included in the span "Amsterdam|". Therefore, this needs to be fixed by adjusting the rules for the SpaCy tokeniser to also split on such symbols. However, also a few cases were detected where such adjustments are not possible and thus these sentences were also removed from the data before using this data as input for all the models. Even if the errors originated from the SpaCy model specifically, it is likely these errors could have an effect on the other models. Hence, these preprocessing steps are also performed for the other models.

6.2 BERT tokenisation

BERT tokenisation uses a hybrid representation (or sub-word representation) of the word-based and character-based representations described in the Literature Study section, which has shown to outperform the other representations [23, 21]. This implies that words will be split according to token splits defined within the tokeniser. For instance, the name "Pieter" can be split up into "Piet" and "##er", since "##er" is a sub-token that is also often used in other words such as "Stater". When a symbol or token sequence is not recognised, it will receive the label "[UNK]". This occurs mostly with the tokenizer from the model BERTje, since it does not recognise symbols such as "€", "%" and "@". Therefore, the BERTje tokeniser is adjusted so that these tokens are added to its vocabulary. Since the RobBERT tokeniser already contains such tokens, nothing was added to the vocabulary of this tokeniser as a result.

Although this hybrid representation is useful for better understanding the shapes and logic of language, it creates a slight issue within the current annotated data, since this data is not annotated according to the hybrid representations used in the BERT tokenizer. There are two main ways to properly solve this issue. The first is to only annotate the first subtoken in a series of subtokens and annotate the rest as a special label such as "-100" to indicate that the rest of this series belongs to this first token. The second option is to label everything in the series of subtokens according to the label specified through the annotation process. For instance, the tokenisation of "['Piet', '##er']" will get the labels "['B-PER', 'I-PER']", which uses the IOB-scheme for labelling the beginning, inside and outside span of the tokens. The latter scheme was used, since it is the most straightforward option when dealing with a NER task and makes interpretability a bit easier.

In addition to the creation of new labels for this tokenisation methodology, extra tokens need to be added that indicate the beginning and end of a sentence. These are the "[CLS]" and "[SEP]" labels for the beginning and end of a sentence respectively and will be denoted by the label numbers 1 and 2. Moreover, padding needs to be added to each of the sentences, since each input needs to be the same size for BERT to work properly. The padding will be assigned to the number 0. All the other labels are assigned afterwards, which creates a total of 33 labels including the "O" (outside) token. These encodings are saved and will be used later when converting the labels of the test set to their corresponding numbers.

6.3 Experiments

Three models were tested for the task of de-identification in this thesis, namely the SpaCy, BERTje and RobBERT model. Moreover, each model will have a maximum three variations, namely base, Regex and CRF. The base variation is the model fine-tuned without any special additions. The Regex model is a system that combines the predictions of the base model with that of a regular expression system to predict all labels. The CRF model uses the logits from the BERT-based models to hopefully find better predictions. Since SpaCy is a different type of model by itself, the CRF model cannot be used on top of this architecture and thus this variation will be excluded in the experiments. Another model that was initially used for this experimentation is Multi-Lingual BERT. However, since it achieved 10% lower scores on average in comparison to the other BERT models, this model was dropped early in the project. This lower performance can be supported by the fact that mono-lingual models outperform the multi-lingual versions in general, as was also noted by de Vries et al. (2019). The hyper-parameters of these experiments were based on the literature and were further fine-tuned

manually. These parameters are shown in the Appendix in Table 17 and are obtained from the corresponding papers [103, 106].

The models themselves were implemented in Python 3. The SpaCy model that is chosen for the current experiments is the large variant, which is the same model that was used for the tokenisation. This same model is used, since this makes its implementation in practice quicker and more simple. Additionally, the large model had the highest scores for the NER task and, due to its increase in size and length in training, it is hopefully able to detect complex patterns much better than its smaller variants. As for the BERT model, two variations were implemented in Pytorch using the HuggingFace library [119], namely BERTje by de Vries et al. (2019) and RobBERT by Delobelle et al. (2020). The reasoning for choosing these two models is shown in the Model section. In addition, different variations of the CRF model were implemented. In the end, the only variant that would work properly with the BERT model is the implementation by Kurniawan (2019), which was compatible with the backwards pass of the BERT model in PyTorch. Therefore, this implementation will be used as a basis for all experiments related to the CRF model.

6.3.1 Regular Expression System

The labels that will be solved through the regular expression system are *EMAIL*, *FILE*, *MONEY*, *POSTCODE* and *WEBSITE*. These labels were chosen, since they each have their own identifier, through which the entire spans can be recognised by a rule/pattern-based system. For this experiment, the system was applied to the train data with all previously considered labels. Through this experiment, it became clear that these specific labels had impressive results, where these labels scored a strict F1 score of 95%. This is much better than what nearly any model could learn given the current labels. Additionally, four out of the five labels are new, which implies that some load is taken off of the models, since it does not have to learn these labels either. The label *MONEY* was also considered, since the models seem to have trouble with separating the label *CARDINAL* from *MONEY*. Additionally, many of such occurrences contain a form of currency, such as the "€" symbol or the "EUR/EURO" combination, which makes them ideal for Regular Expression. A similar reasoning can be used for the other labels, where the *EMAIL* label always consists of a "@" symbol, the *FILE* label various file formats, *POSTCODE* a combination of 4 numbers followed by 2 letters and *WEBSITE* having a "http(s)" and specific ending formats. However, given the diversity of language used in these documents, some specific changes needed to be made when considering the previous Regex table (Table 13).

First, the *MONEY* label has many variations when considering the format of a "€" type pattern. It can happen that the "€" symbol occurs before or after the money amount. Moreover, the "EUR/EURO" pattern does not only occur before or after the money amount, but also varies in terms of length and capitalisation. In addition, the format of the amount can also change drastically, where the ending of an amount can be equal to for instance ",–'', ",-'' or " -''. In order to handle these variations, multiple patterns were created for this label specifically. Since one pattern is more general than the other, the order of these patterns is extremely important. Therefore, the more specific patterns should be used before the more general ones. This can be seen in Table 18, where first the more specific patterns are displayed before the more general ones. It is also important to note that all these patterns do not take priority over the labels predicted by the model. This implies that, if the model predicted the *CARD* label, the predicted *MONEY* label will be discarded. This rule holds for the all other labels except the *FILE* label, since this label specifically can contain other entities as well. Therefore, these entities need to be discarded in favor of the *FILE* label. For instance, if the file name is as follows "bankname1_person1.pdf", then this should be labelled as a file and not as a bank and person-based entity.

The second change in the system is in regards to the *POSTCODE* label. Something that would occur rather regularly was that the label would identify patterns, where the year number or a similar format would be followed by a Dutch 2-letter word. For instance: "Het jaar 1998 is een .....", where the regular expression would find "1998 is" and consider it a postal code. Since there are only a limited number of 2-letter Dutch words, all of these words were searched and were put in a lexicon. If this pattern would find the "is" in a sentence and it is not capitalised and it contains a whitespace, then the system now assumes that this is not a postal code. This makes the pattern more conservative and more accurate as a result.

A third change includes a second file-based pattern that includes whitespace. This is normally not possible, since it is tedious to find files including whitespace in normal text without any type of indication. However, if the document is an email, the "Attachments" phrase is used, through which more complex patterns for the label *FILE* are possible. Other than these changes, some minor changes also occurred, where certain file formats and website ending formats were updated. This is something that needs to kept track of if this system is used in practice.

The order in which these patterns are used is also crucial for making this system work properly. First, the "Attachments" check is performed for the label "FILE" to ensure that no errors can be made by putting this check later. Next, the *EMAIL* patterns are checked before the *WEBSITE* label, since the *WEBSITE* pattern contains many elements similar to that of the EMAIL patterns. The only difference is that emails should always have the "@" symbol in their pattern. As a result, the *EMAIL* label patterns are more strict than the *WEBSITE* patterns and thus need to be checked first. Afterwards, the rest of the label checks follow. Similar

to the previously mentioned pattern recognition steps, the *MONEY* label follows a similar methodology, where the most strict patterns are put first, while the less strict patterns follow afterwards. This entire system and their ordering is shown in the Appendix in Table 18.

6.4 Evaluation metrics

Two main types of scores will be used to evaluate the NER models tested in this thesis. The first metric will be referred to as the "Strict Score". This is a score that is often used for the NER task and was utilised for various CoNLL tasks. This strict score implies that both the span and label of the predicted entities need to exactly match the true entities. If both of these requirements are met for a single entity, then the predicted entity is correct and will be counted as a true positive. If there is a predicted entity that does not follow either of these requirements, then it will be seen as a false positive, which will be used to calculate the Precision metric. If there is an entity that has not been detected by the model, then these entities are seen as a false negative and thus are used to calculate the Recall metric. This score is fully defined in the first item of List 6.4.

The second metric is a new metric that was created based on the task of anonymisation. The name for this metric is the "Relaxed Score" and its requirements are shown below in List 6.4. Note that the letter behind each requirements shows to which score this requirement belongs. In other words, all requirements are related to the relaxed score and if either of these requirements is met, a predicted entity will be seen as correct. In contrast, only the first requirement belongs to the strict score and only then will a predicted entity be seen as correct. This new metric was created, since the label and exact span are not necessarily important for anonymising documents. Instead, the importance lies in locating privacy-sensitive information and ensuring that the entirety of the true entity is included in the prediction as was discussed in the Literature Study section. Additionally, the label is also of no importance for this task, since anonymising the documents is not reliant on the label. Therefore, it would be interesting to consider how accurate the models are when only considering the entity spans from the predictions or by combining multiple predictions into one. Note that the requirements below calculate the micro score for each type of score respectively.

S+R: The span of the predicted entity and the label must exactly match the span of the true entity. In mathematical definition: the prediction is correct if and only if $s_{true} == s_{pred}$, where s_{true} represents the true entity/sequence of tokens and s_{pred} the predicted entity/sequence, where each of the individual elements c_s, c_e and l need to be equal within these sequences. The Literature Study section provides more general definitions on the letters and symbols used in these mathematical definitions.

R: If the span of the true entity is being contained by a predicted entity, then it is seen as correct, even if the labels do not match. Additionally, if more true entities are contained in this span, than these will also be seen as correct. This also means that the labels do not have to be correct in order for this prediction to be true. In mathematical terms: Given a predicted entity s_{pred} with span $(c_{s,pred}; c_{e,pred})$, the prediction(s) is/are correct if $c_{s,pred} >= c_{s,true}$ and $c_{e,pred} <= c_{e,true}$ for any true entity $s_{true,i}$ in entity sequence S_{true}.

R: If multiple predicted entities are located directly next to each other and these entities form a span in which a true entity is contained, this is also considered as one correct prediction. The predictions combined will be seen as one prediction and will thus not be considered further when evaluating the rest of the sentence. This can only be true if each neighbouring entity is separated by a whitespace or similar separation symbol. Additionally, this reasoning will not hold if the prediction overlaps with another entity, since this could compromise another correct prediction. The labels also do not influence this requirement similar to the previous requirement. In mathematical terms: By combining multiple entities $s_{i,pred}$ into one entity s_{comb}, with *comb* equal to set $..., i-1, i, i+1, ...$ of length *N* and with each element of this set corresponding to the requirement $c_{e,i-1} + 1 == c_{s,i}$, the true entity was correctly predicted if $c_{s,comb} <= c_{s,true}$ and $c_{e,comb} >= c_{e,true}$.

For each of these scores, various metrics will be calculated. There are two main ways to calculate the scores for each of these metrics. The first is to use the IOB-scheme and use multi-classification to detect the entity and non-entity tokens, as is mostly done with BERT. The second option is to consider the position and label separately as is generally done by SpaCy using the definitions given in the Literature section. Since this dataset was mostly annotated with the SpaCy tokeniser, the data was saved in this same format. As a result, the second option is chosen for calculating each of these metrics. Moreover, in this way, the relaxed score is easier to calculate for each of the metrics. Although this may seem to create problems at first for existing metrics, the general definitions for Precision, Recall and F1 hold even with the change in requirements listed for the strict and relaxed score. Therefore, these metrics will be used to evaluate the performance of the models given the new requirements. For this performance, both the micro and macro scores will be considered. Although both will be considered, the micro score is considered the most important, since not all labels should have the same weighted value. The label for which this weighted value is different is the *MISC* label, since it encompasses all miscellaneous entities, which is much less important than for instance the label *PER*, *GPE* or *MONEY*.

Another metric that will be considered for evaluating the models is the number of strict and relaxed sentences predicted correctly. This is done to give more insight into the performance of the models on a sentence level, since all the previous scores are created on entity level. The "Sentence Score" is a score that indicates what percentage of all sentences in the test set were predicted completely correct. A sentence is completely correct if there are no false positive and false negative errors for this sentence given the predicted entities by the models and the true entities in the test set. The percentage of sentences that are correct is then the number of correctly predicted sentences for each respective score divided by the total number of sentences in the test set. In mathematical terms: A sentence is correct if $S_{pred} == S_{true} \; \forall \; s_i$ and s_j in sequences S_{pred} and S_{true} respectively. Thus, this score can give an approximation of how many sentences are correctly de-identified in the test set for each of the score types described previously.

7 Results

7.1 Training and Validation

The main results from the experiments described in the Set-up Experiments section are shown in Table 5. The training and validation results for both the best SpaCy and BERT-based models are shown in Figure 14 and Table 4 and will be discussed first. Before discussing these results, it should be noted that the models are trained using the strict score, which biases the models generally towards this score in contrast to the relaxed score. Additionally, it should be noted that the training of both BERTje and RobBERT is based on a token-level output and not on an entity-level output. Since the traning of these models is performed on a token-level, the training scores are distinct from the SpaCy training scores and thus cannot be compared. This is also the reason why only the strict scores are used for these training statistics. Additionally, the RobBERT model showed similar training and validation scores to BERTje. Therefore, these scores will be shown in the Appendix in Table 19 and not specifically mentioned further here. Despite the differences in training and validation scores, it is evident that the overall strict test scores in Table 5 seem much lower compared to the training and validation scores shown in their respective figures. There are multiple justifications for this phenomenon.

First, it should noted that, although the test set somewhat resembles the training set, there are also different document types included in the test set to help check for generalisability. Therefore, this discrepancy might be caused by the lack of generalisability to new documents. This will be thoroughly checked in the Error Analysis section. Second, the model could be overfitting on the training set, which would explain the much lower strict test set scores. However, after carefully considering this option and observing the training procedures, this does not seem to be the case. This is supported by the continuous and gradual increase of both the strict train and validation scores shown in Figure 14 and Table 4. In contrast, the relaxed scores in the SpaCy graph only show a small decrease at the fifth epoch, which continually increases after this epoch again. As for the BERTje model, both the train and validation scores continually increase, although the validation score experienced a large increase in the fourth epoch. This indicates that either the models could be trained for longer to obtain even higher scores or shows the instability of the training for the BERT models. However, after performing an extra experiment, where the number of epochs has been increased, the observed differences in results and scores were small and, to prevent overfitting on the training data, the number of epochs for the SpaCy and BERTje model was thus limited to 10 and 4 respectively. A third explanation for the difference in scores is that the models have trouble learning due to the difficulty of the task at hand. Its difficulty originates from both a new domain on which these models are fine-tuned and the addition of new labels, which makes it challenging for pre-trained models to learn the data properly. This is a reasonable assumption, since the datasets on which both the SpaCy and BERT models were trained are vastly different from the current dataset used within the experiments, as was discussed in the Model Descriptions section. Additionally, the BERT model can be used for multiple purposes and hence its focus is not necessarily on NER, which could explain its instability during the training process.

7.2 Model Scores

As is depicted by the bold percentages in Table 5, the SpaCy model with the Regex system has the best overall results, with the Strict F1 score being the highest of all models and their variations. The F1 score of this SpaCy model comes closest to the scores within the literature and has a minimum difference of 4% compared to the other models from the experiments. This could be a consequence of the inclusion of various features for the NER SpaCy model, which likely helped identify the context of the entities better, as was described in the Model Descriptions section. Since the BERT based models do not include such features, it is more difficult for these models to consider the context of the entities. Despite it having the largest F1 score, the highest strict recall score was obtained by the BERTje model with the Regex CRF variation. In general, it can be observed that the recall scores for de BERT-based models are higher than those of the SpaCy model. Normally, the recall is the more difficult score to improve than the precision as is the case with SpaCy. However, as will be shown in the Error Analysis section, this is most likely due to the more greedy approach of the BERT-based model when compared to the more conservative approach by the SpaCy model and is thus not caused by some error.

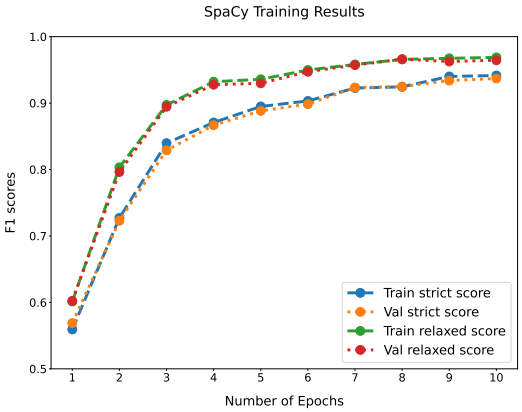

Figure 14: The training and validation results using the strict and relaxed scores for the SpaCy Regex model

Table 4: The training and validation results for the BERTje Regex CRF model.

	F1 Train	F1 Validation
Epoch 1	0.966	0.910
Epoch 2	0.984	0.915
Epoch 3	0.991	0.918
Epoch 4	0.994	0.929

Similarly, the SpaCy model outperforms the other models when considering the macro scores. These scores are quite a bit lower than those of the micro scores. However, this is most likely caused by the *MISC* label, which has an F1 score of approximately 30%, which will also be further discussed, analysed and compared to the other labels in the Error Analysis section.

When considering the relaxed scores, the BERTje Regex CRF variant seems to outperform the other models. As was previously mentioned, the BERT models seem to be much more greedy with their predictions in contrast to the SpaCy model. As a result, it manages to pick up on more entities than the SpaCy model as the scores indicate and is thus the best choice for anonymising documents if the strict boundaries and context are not important for later purposes. In other words, it seems that the BERT model excels at the syntactic task of NER, while SpaCy is better at the semantic part of the task. Additionally, the BERTje Regex variant slightly outperforms the CRF model when considering the Sentence Scores. Notable is that the strict sentence scores are higher than that of the F1 scores. This implies that, if there is an error in a sentence, this effects other entities and predictions in this same sentences, which can cause multiple errors. Additionally, it can also be caused by the fact that sentences with no entities are included in the test set, which are generally much easier to discern than the sentences with entities. In contrast to the strict score, the number of correct relaxed sentences is lower than that of the relaxed F1 score. This is what would be expected, since one sentence can depend on one or more entities and hence is generally more difficult to get higher scores for.

Another notable observation is that the regular expression systems works wonders when combined with the base predictions by the models. As was discussed in the Set-up Experiments section, some labels were predicted with the regular expression system and received high scores, which are reflected by all models. RobBERT is the model that benefits the least from this increase, since it probably has a more extensive tokenizer for symbols such as the "@" and "€" sign, which makes its base predictions better in comparison to BERTje model. In addition, the architecture used in RobBERT is the large version of the BERT model and thus could help with the more complex predictions. However, this made the model also more difficult to train and, given the difficulty and instability of this training process, it could have resulted in worse predictions overall. When comparing BERTje and RobBERT during training, BERTje was able to converge faster and gain better scores than RobBERT when using the regular expression system. Moreover, it seems that BERTje is able to better adjust to labels such as *PERSON* or *ORG* than RobBERT. This can be also inferred from the scores mentioned in the original RobBERT paper [106], since BERTje seems to outperform the RobBERT model on the CONLL 2002 task, which contained such labels. However, when observing the base models, RobBERT does seem to form a better basis for the predictions than BERTje without the Regex system. Therefore, if this Regex system is not an option, RobBERT would be preferred over BERTje, given both the strict score and relaxed score differences.

The CRF model seems to have little influence when considering the overall scores. Generally there is a 1% increase in score when using the CRF model in comparison to the Regex model variants. This is likely caused by the fact that the CRF model was not pre-trained for this specific task and thus not working as well as initially expected, even if it did help the overall scores a bit. Another cause for this could be that BERT by itself already manages to find many of the entity spans by itself, which makes the CRF less effective for BERT than for other models, such as the BI-LSTM as was shown in the literature [23]. A different cause would be that, despite its improvements, that the backwards pass did not work as well as was intended.

Table 5: Results from the main experiments."Rel." implies the relaxed score, "Prec." implies the precision and "Sents" indicates the Sentence Score

Model	**SpaCy**		**BERTje**			**RobBERT**		
Type	**Base**	**Regex**	**Base**	**Regex**	**Regex CRF**	**Base**	**Regex**	**Regex CRF**
Strict Prec.	75.00%	**87.02%**	67.09%	77.18%	78.41%	71.30%	77.02%	77.19%
Strict Recall	75.22%	81.61%	74.32%	81.74%	**82.65%**	74.59%	78.41%	80.03%
Strict F1	75.11%	**84.23%**	70.52%	79.39%	80.48%	72.91%	77.71%	78.59%
Rel. Prec.	83.17%	**91.57%**	77.91%	90.13%	91.46%	81.78%	89.79%	90.32%
Rel. Recall	82.02%	86.19%	86.31%	92.69%	**93.76%**	85.55%	89.36%	91.91%
Rel. F1	82.59%	88.80%	81.89%	91.39%	**92.60%**	83.62%	89.57%	91.11%
Macro Prec.	73.56%	**84.35%**	63.99%	76.49%	78.41%	67.18%	75.30%	76.33%
Macro Recall	72.26%	78.46%	68.92%	79.74%	**80.28%**	68.92%	76.06%	77.37%
Macro F1	72.60%	**80.94%**	66.14%	77.95%	78.95%	67.95%	75.61%	76.69%
Strict Sents	80.38%	84.89%	80.41%	84.93%	**85.26%**	81.84%	83.98%	84.45%
Rel Sents	82.84%	87.64%	86.68%	89.16%	**92.23%**	87.51%	87.36%	91.41%

8 Error Analysis

This section will go into detail on the errors that were made by the models. It should be remarked that not all models will be considered for this error analysis. Only the best variation for each model type will be discussed and further analysed in this section, since these are the models that will be considered for practical use. The models considered for this analysis are: "SpaCy Regex", "BERTje Regex CRF" and "RobBERT Regex CRF". Additionally, the initial intention of this section was also to perform a word-based explainability analysis of the NER models. However, due to the nature of the CRF implementation and the usage of SpaCy, it is not possible to extract the prediction probabilities for each label. Hence, this will be left to future work. First, the distribution of the errors across the labels will be discussed for both the strict and relaxed metric. Afterwards, the generalisability of the models will be analysed by considering two new test sets that concern new documents and new entities.

8.1 Label analysis

8.1.1 Strict score errors

The Figures 15 and 16 depict the distribution of the false positive and false negative errors across the fifteen labels using the strict score. Figure 15 gives the total number of errors per label for each of the models, while Figure 16 shows the total number of errors with respect to the total number of entity occurrences for each given

label (*fp / total* and *fn / total*). It is observed in Figure 15 that the labels predicted with the regular expression system have the lowest number of errors, which is as expected. Of these labels, the label "MONEY" has the largest number of errors, which is a consequence of this label having the most variation in terms of the Regex patterns and largest number of occurrences. For most of these labels, the number of false positives entities is lower than that of the false negatives. This is a consequence of the regular expression patterns missing some specific entities, which is common in such systems [24, 23]. In addition, in terms of error amount, the labels *GPE* (Geo-Political Entity) and *TIME* also seem to perform particularly well in both plots. This could be caused by the fact that these labels have been used before in all models. Moreover, the format of such labels does not change as frequently, which helps the models predict such labels particularly well. Additionally, the number of entities having such labels is relatively small compared to other labels such as *DATE* and *ORG*.

In contrast, most of the errors seem to occur for the label *ORG*. This is partially caused by the the relatively large number of organisation labels present within both the training and test set, as is supported by Figure 7 in the Annotation section. However, this cannot be the only cause, since the errors for this label generally appear to decrease in Figure e16 while still having largest number of errors. Hence, another cause of this could be due to the amount of variation within this label. Due to this variation, it is possible that the model has trouble deciding on the boundaries for some of these companies, which will be shown later in the Generalisability Models subsection. Another cause for the large amount of errors for this label could be the pre-trained weights for these entities. When considering the fact that other domains were used for training the NER models, it is also likely that the current pre-trained models do not recognise most of the organisations shown in the financial domain. Hence, it would be interesting to investigate whether a new label for organisation names would improve performance.

The distribution of the other errors seem rather well balanced. However, it can be deduced from Figure 16 that the *MISC* label in particular shows a large increase in terms of errors when compared to Figure 15. As was previously discussed, the *MISC* label is the least important of the labels and contains the most variation in terms of entities. Therefore, neither model has properly learned this label. Additionally, its usage in this model is somewhat different from that of the previous literature-based tasks. The label *STREET* does not seem to perform that well either. It is highly probable that the models were also not able to learn this label as well due to the relatively small number of entities for this label. Moreover, the *STREET* label is a new label to be learned by all models and made it thus more difficult to learn such entities, especially when considering that such labels also have some overlap with the *CARDINAL* label. For instance, consider the following entity: "Streetname1 55". There are two methods for annotating this entity, namely marking both the streetname and the number as the label *STREET* as seeing this as one entity or marking only the name of the street as a *STREET* label and the number specifically as a *CARDINAL* label. Although neither can be considered incorrect, the former method was used during the annotation process, since, in this way, other numbers can be differentiated from street numbers. However, the latter occurred more often in the BERT models specifically, as is also shown in Figure 16. Therefore, it is plausible that the models are confused about this distinction.

When comparing the models by themselves, it is reasonable to state that the SpaCy model in general outperforms the other models in terms of strict errors, which is supported by the scores shown in Table 5. Specifically, SpaCy has been able to learn the labels *DATE*, *GPE*, *ORG*, *PER* and *STREET* reasonably better than the BERT-based models. There may be multiple general reasons behind its better performance on these labels. First, as was also mentioned in the Results section, SpaCy is much more conservative with its predictions. As a result, this model does not have as much false positives for most labels when compared to the BERT models, which improves its scores. Additionally, most of these labels have been pre-trained by the SpaCy NER model and, as a result, it could be that SpaCy therefore adapts generally better to new entities with these labels than the pre-trained BERT models. Third, the BERT models are general models trained for multiple purposes, while only the NER component of the SpaCy model is updated. Hence, this could influence how well each model performs on the strict score specifically.

However, in contrast, the BERT models do sometimes outperform the SpaCy model for some of these labels. For instance, the labels *CARD*, *DEP* and *ID* are arguably better than that of the SpaCy model due to the much lower number of false negatives for these labels. This could be a consequence of these labels being easier to predict in text, since most texts consists of letters and not numbers. In addition, since the BERT-based models seem to be much more greedy with their predictions, it is likely that its predictions of specifically the labels *CARD* and *ID* are due to its easier detection in text. However, this reasoning does not hold for the *DEP* label, since such entities generally do not contain numbers generally. Its arguably better performance of the *DEP* could also be caused by BERT being able to learn specific words or patterns better than SpaCy and being able to detect such entities with more ease.

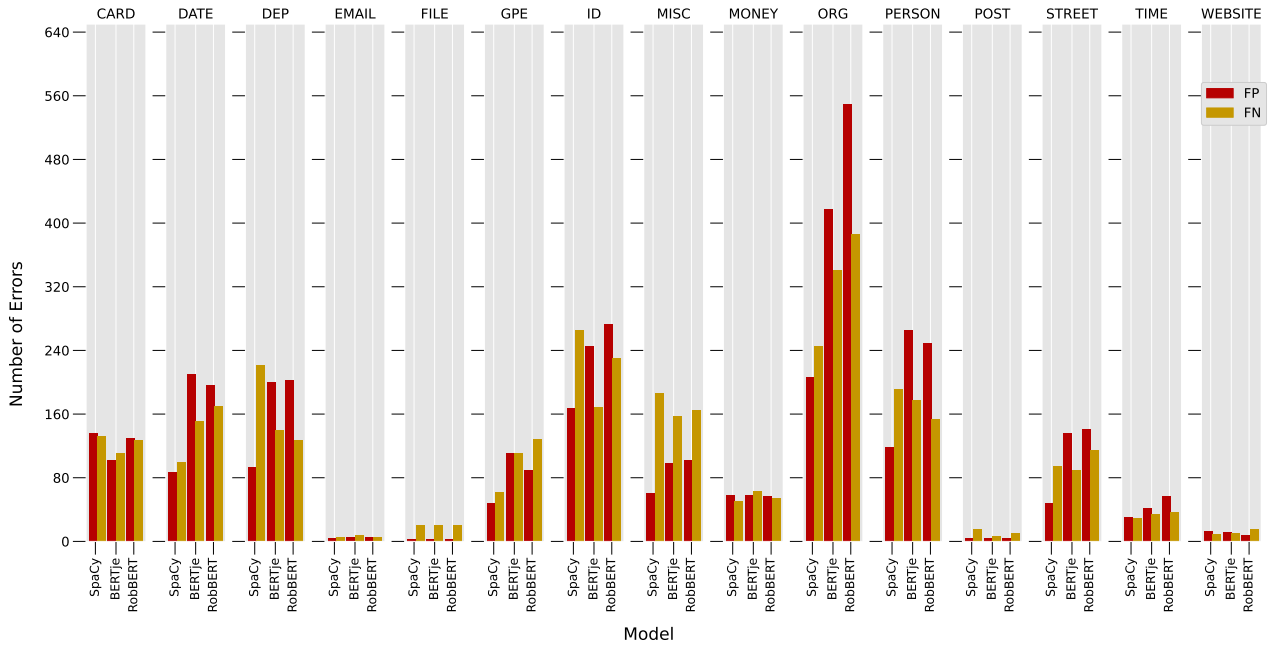

Figure 15: The number of false positive and false negative errors for each label and model evaluated using the strict evaluation score.

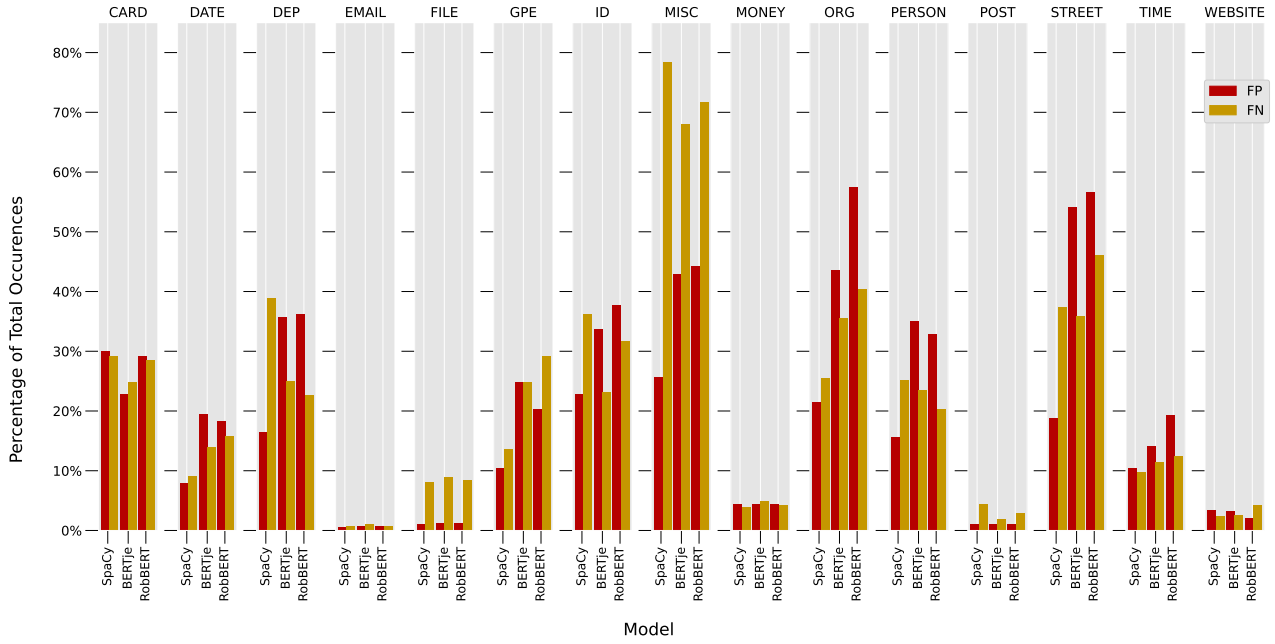

Figure 16: The ratio between the number of false positive/negative errors and the total number of entities annotated in the test set for a given label. This is compared for each label and model with respect to the strict evaluation score.

8.1.2 Relaxed score errors

Two more graphs are created for the relaxed score, which are shown in Figures 17 and 18 and resemble the same idea as the graphs from the previous section. Before considering the shape of the graphs, it should be noted that there are much less errors in these new figures than in the previous Figures and hence the y-axis for the total number of errors has changed. This is due to the overall higher scores for the relaxed metric as depicted in Table 5. Therefore, this should be taken into consideration when investigating the shapes and number of errors within these figures in comparison to the previous statistics.

When examining Figure 17, the first detail noticed is the large false negative error for the *DEP* label for the SpaCy model. This implies that, for many sentences in the test set, SpaCy was not able to identify the department in the given sentences. Additionally, it is important to consider that the number of errors for this false negative score has only slightly decreased in comparison to that of the strict score as shown in Figure 15. Moreover, it seems that the labels *STREET*, *PER* and *MISC* also experience this same phenomenon but to a somewhat lesser degree. Since these large error deviations are somewhat unexpected, the predictions were investigated manually and some example sentences containing such errors are shown below in Table 6. Note that all privacy-sensitive information has been de-identified, but should not further impact the quality of these examples. Additionally, only the SpaCy model was considered, since this phenomenon does not occur for the other models. Moreover, after investigating some of the errors made by the other models, no notable or different errors or patterns could be identified for any of the labels. Hence, examples of other models will not be included further in this error analysis.

The first *DEP* sentence for the SpaCy model shows an error where only the word "Stater" is being recognised by the algorithm, both as an identification number (*ID*) or organisation (*ORG*). Therefore, the "Department name 1" has not been recognised by the algorithm. Additionally, seeing the word "Stater" as both an identification number and organisation is also incorrect. These errors could be caused by an insufficient amount of training data or fine-tuning for the *ID* and *DEP* labels. Moreover, it could be that some consistent errors were made in the training data, which would have made these categories more difficult to learn. Additionally, the test set includes many instances of similar sentence, which increases the number of errors further. Furthermore, many of these sentences are duplicates, which makes these errors also occur quite often for specific department names. It is also noticed that the algorithm has trouble deciding boundaries for these entities as well, where they only manage to detect a part of the entity but miss the bigger picture. This is also an error that occurs frequently for the BERT models. However, this is due to these models predicting too much.

The *STREET* label is another occurrence of a category that has a much higher false negative rate for the relaxed score in comparison to the strict score for the SpaCy model. However, the majority of these false negative errors are caused by street names simply not being recognised. An example of this is shown in Table 6. Additionally, there were many duplicate sentences of these errors as well, which made this number increase drastically too. In contrast, the *MISC* label has few duplicates but also has the problem of only being partially recognised. This is shown in the same table, where specific parts of long entities are simply not recognised. Despite the rather small change in these false negative errors, this only seems to be the case for the SpaCy model. Therefore, it appears that the more greedy approach by BERT, despite it not working too well for the strict score, seems to pick up on more of such entities that the SpaCy model. Based on these observations, it is evident that the SpaCy model does not benefit as much from the relaxed score compared to that of the BERT models, which is supported by the main scores shown in Table 5.

It is also noticed that specific labels improve more than others for the relaxed score. The *STREET* label is a great example of this, since it helps fix a main issue that the BERT model had with this new label. As was previously discussed for the entity "Streetname1 55", there can be confusion between two types of annotations when regarding this label. Since this was especially true for the BERT models, the number of errors has decreased significantly. Another label that works much better with this rule is the *DEP* label, since it manages to fix a similar error. Consider for instance, "Stater Department team Stater", which was annotated to be a organisation-related entity (department). However, BERT again makes two predictions, where it separates this full entity into 2 smaller ones. Therefore, still the full entity is anonymised. As a result, the relaxed score counts this as correct. A similar phenomenon was discovered for the label *PERSON*. In addition to these improvements, it also becomes clear why BERTje outperforms the RobBERT model. The labels *DATE*, *ORG*, *PER* perform better in general for the BERTje model, which are also the labels that occur the most. Therefore, even if RobBERT is able to outperfrom BERTje on the label *DEP* for instance, BERTje is still preferred over RobBERT given its better performance on the most frequent entities.

Table 6: A table containing some incorrectly predicted sentences from the test set considering the relaxed score with the SpaCy model.

Type & #	Sentence	Pred	True
DEP 1	“To: Stater Departmentname1 Stater <departmentname1@stater.nl>,”	[(4,10,‘ID’), (26,32,‘ORG’), (33,64,‘EMAIL’)]	[(4,32,‘DEP’), (33,64,‘EMAIL’)]
DEP 2	“To: intermediair desk <intermediairsdesk@bankname.nl <mailto:intermediairdesk@banname.nl> >”	[(22,51,‘EMAIL’), (52,91,‘EMAIL’)]	[(4,21,‘DEPARTMENT’), (22,51,‘EMAIL’), (52,91,‘EMAIL’)]
STREET 1	“Adres Streetname1 88 1169 AA Amersfoort”	[(20,27,‘POST’), 28,33,‘GPE’]	[(6,19,‘STREET’), (20,27,‘POST’), 28,33,‘GPE’]
MISC 1	“/MEA/EAC/ORG limited/IMRE/UI1/983921 LENING 888.111 UITKERINGOV”	[(4,24,‘ORG’), (46,53,‘ID’)]	[(0,65,‘MISC’)]
MISC 2	“UU ST ERR 8782112”	[(3,9,‘MISC’), (10,17,‘ID’)]	[(0,9,‘MISC’), (10,17,‘ID’)]
PER 1	“G.T.L. de Kleine (BSN: xxxxxxxxx) Uitloggen ? ()”	[(0,9,‘PER’), (23,32,‘ID’)]	[(0,16,‘PER’), (23,32,‘ID’)]
PER 2	“Persoon : de Klein-Vermeulen”	[(10,18,‘ORG’), (19,28, ‘PER’)]	[(10,28,‘PER’)]

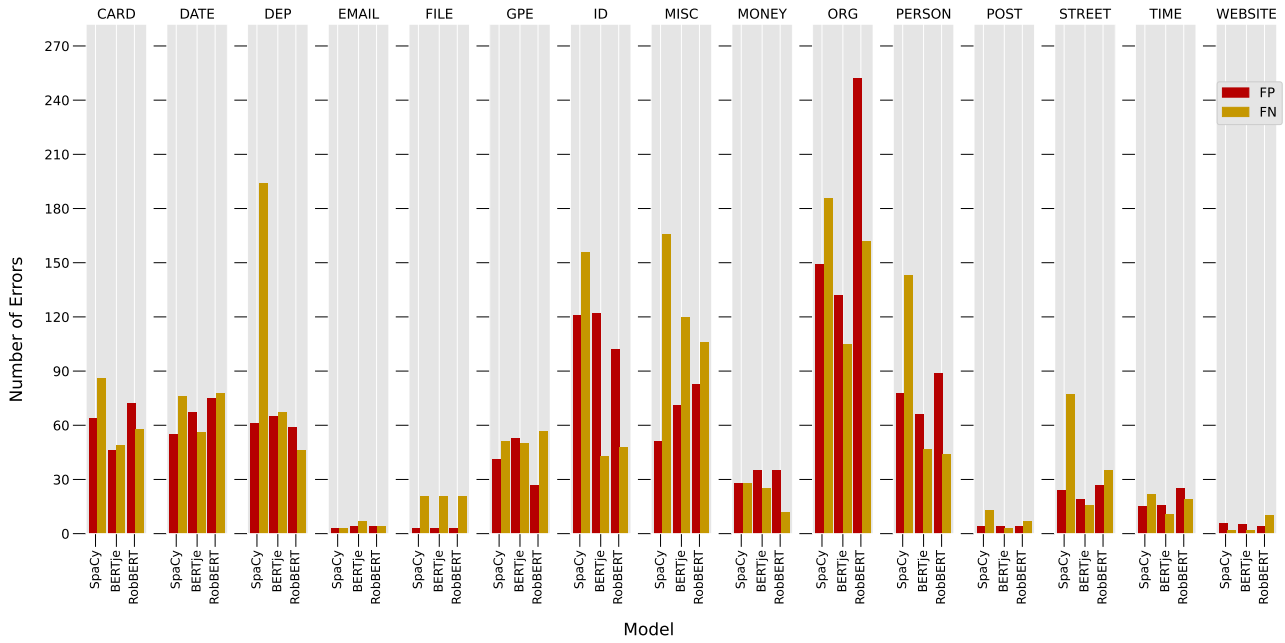

Figure 17: The number of false positive and false negative errors for each label and model evaluated using the relaxed evaluation score.

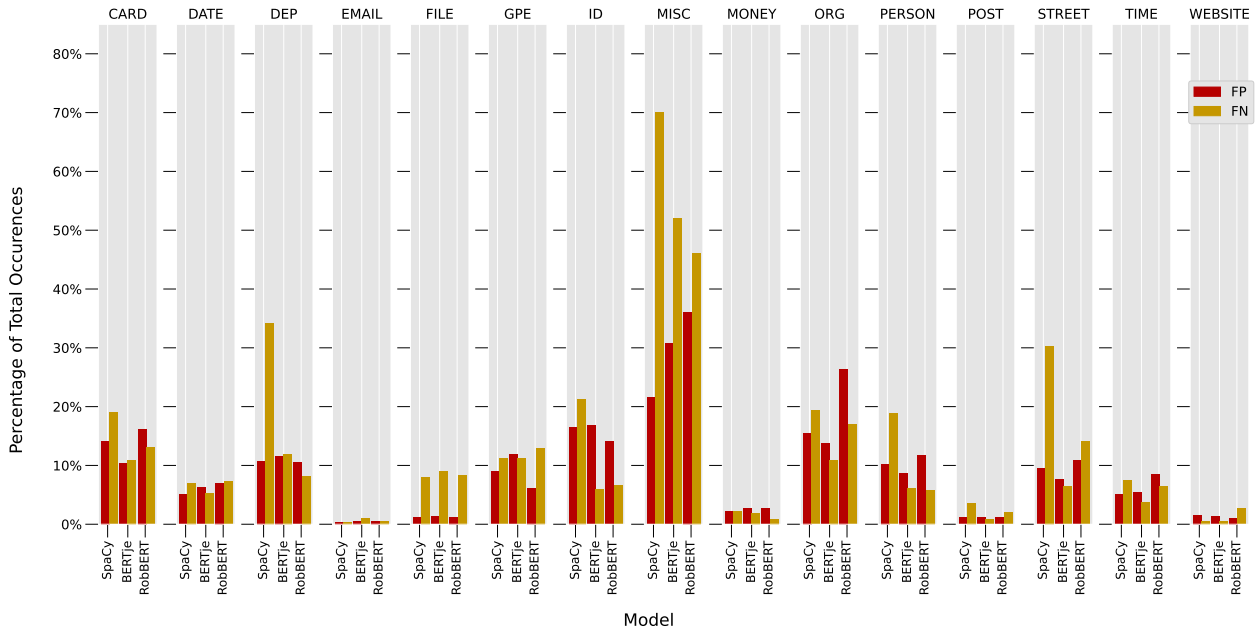

Figure 18: The ratio between the number of false positive/negative errors and the total number of entities annotated in the test set for a given label. This is compared for each label and model with respect to the relaxed evaluation score.

8.2 Generalisability Models

8.2.1 Set-up Generalisability Experiments

To test the generalisability of the best models, two more experiments were performed. The first is an experiment where only new documents will be considered for the test set. Since the intent was to only use emails to train the models, all emails were used during training. Therefore, only certain directories from the inquiry dataset were considered for this analysis. These directories are provided in the list below and more general statistics on these directories are shown in Table 12 in the Appendix. The documents in these directories provide a total of 1151 sentences, which is about $1/8$ of the full test set. This number should be sufficient to judge the generalisability of the model on relatively new and unseen inquiry documents.

- "Offerte iets"
- "Eigenaarsinformatie"
- "Afschrift betaalrekening details bijschrijving"
- "Schenkingsovereenkomst"

The second experiment concerns the generalisability of the models towards new entities. In order to determine which entities are new, first all entities that are contained in the training set are not considered for this new set. However, this is not sufficient, since there can be many variations of the same entity. For instance, "Stater Department1" and "Stater_Department1" should be considered the same entity. To avoid most of these biases from the training set, the Levenshtein Distance is used, which calculates the minimum number of single-character edits that are required to change one of these words to the other. Through this metric, words that are too similar will be removed. This is done by first converting the number of edits to a ratio and afterwards deciding on a threshold. For this experiment, the threshold is set to 90%, since only entities that nearly perfectly resemble entities in the training set need to be removed. However, this threshold will not be applied to labels that contain mostly numbers, which are *CARDINAL*, *DATE*, *ID*, *MONEY*, *POSTCODE* and *TIME*, since it is assumed that each variation for each of these labels is a separate entity. For instance, if there is a loan number "5432212" and there is another number "5432202", they are considered distinct numbers and thus it would not make sense to use this threshold for such categories. After applying this threshold to the text-based entities and combining them with the entities of all number-based categories, only the sentences containing these unique entities will be considered for this experiment. Therefore, all other predictions not related to these unique entities will be removed. After performing this preprocessing, this set contains a total of 2156 sentences and new 3,299 entities, of which 2268 entities are unique, which should be a sufficient amount of data to test on. The distribution of the labels for both experimental sets are shown in Table 7.

Table 7: The distribution of the labels within the experimental sets.

	CARD	DATE	DEP	EMAIL	FILE	GPE	ID	MISC
New Doc set	66	230	11	0	0	62	170	46
New Ent set	218	575	26	151	13	90	690	36

	MONEY	ORG	PER	POST	STREET	TIME	WEB	Total
New Doc set	492	115	93	31	27	17	9	1,309
New Ent set	1087	95	468	139	27	186	55	3,299

For both of these experiments, the strict, relaxed and sentence scores will be shown. However, the false positives and false negatives will both not be shown, since they are relatively consistent with the F1 scores shown by the results. In addition, the macro scores will not be included in this evaluation, since the frequency of new entities differs vastly per label. This imbalance is shown in Table 7. From this table, it becomes evident that some labels do not occur at all for the unique documents set. This a consequence of the first experiment missing any email data, which covers labels such as *EMAIL* and *FILE* to a much greater extent, as is supported by Table 6 in the Annotation Process section. However, these labels are solved by using the regular expression system and are thus not as crucial for this evaluation. In contrast, the labels *ID*, *MONEY* and *PER* occur the most often, which is consistent with the statistics for the inquiry documents depicted in Figure 6. Intriguingly the same can be said for the unique entities set, where these labels also occur the most. The reasoning behind this similarity is the fact that the Levenshtein similarity score was not used on the number-related entities, which makes them appear more often than the other entities. Additionally, persons often have different name formats and thus also occur more often.

8.2.2 Results and Analysis

The main results of both experiments are shown in Table 8. From this table, it becomes evident that the models still perform relatively well on new documents and entities, although the scores have been lowered a bit. Overall, it seems that the strict scores for all models have been lowered by 2-3%, while the relaxed scores are lowered by 1-2%, which are still respectable results. Due to the decrease in the strict score, the SpaCy and BERTje model now seem to be on more equal terms. BERTje even manages to outperform the SpaCy model given the unique documents experiment. This could imply that the SpaCy model is able to perform better on the emails rather than the inquiry dataset. Furthermore, the RobBERT model now slightly outperforms the other models with the relaxed F1 score. This could be due to mere coincidence, since the difference in score between BERTje and RobBERT is negligible. As for the other metrics, the highest scores seem to occur at the same location as in Table 5. In contrast to the strict and relaxed scores, the strict and relaxed sentence scores have been lowered by 4-12%. The largest decreases in these scores are observed for the new entity set. This is a consequence of only considering the sentences that contain the new entities identified through the pre-processing steps, thus also removing the easier to predict sentences that did not contain any entities.

When investigating the difference between the document and entity experiments, it seems that the entity experiment overall provides better results. For both the SpaCy and RobBERT model, it seems that the entity scores are higher than the document scores. This can be explained by considering the larger number of entities within this new test set, which could make the predictions better overall. Although this trend also holds for BERTje when considering the relaxed score, the strict document scores for this model seem to outperform the strict entity scores. This discrepancy can have multiple causes. First, the document set has more *ORG*, *MISC* and *GPE* entities relatively speaking in comparison to the unique entity set. Since these labels are generally more difficult to predict and the *ORG* label occurs rather frequently, it could explain the lower overall results for the document-based experiment. Second, it could be that the models are able to generalise less well to the inquiry documents, since the entity experiment contained both information from the email and inquiry dataset, while the document experiment only contained inquiry-based entities.

Since these scores only provide general results, it was also checked how well the labels performed individually. These graphs are shown in Figures 19 and 20. In Figure 19, it is observed that the email and file scores are at 100%. This was done, since such entities were not detected within these inquiry documents, which also implies no false positives were detected. Although most labels perform as expected, there are some observations to note. First, the categories *CARD* and *DEP* seem to vary the most for the models in terms of performance. The

DEP label most likely varies this much, since only 11 entities of this label were detected within the document, which makes the results somewhat unreliable. As for the *CARD* label, this is more difficult to determine. However, as can be deduced from the label error graphs previously discussed, SpaCy did have the most trouble detecting *CARD* labels in terms of errors. Therefore, its generalisability of this label to new entities is most likely also compromised. Moreover, it can be seen that the BERTje model outperforms the other models, which could also partially explain its lower number of errors in the previous graphs. Another finding is the relatively low strict score for the *WEBSITE* label. However, this is also due to this label occurring very rarely in these documents. In contrast, the labels *DATE* and *TIME* seem to perform better than expected, since they are not solved through regular expressions. This may be due to the similar formats of these entities within these documents specifically. It is also observed that BERTje and RobBERT both are able to detect the *TIME* label better, while the SpaCy model has learned the *DATE* label better. Furthermore, it is also observed that, for the *GPE*, *ID*, *ORG* and *STREET* label, that the BERT-based models pick up on more of such entities than SpaCy, which is supported by their higher scores in both Tables 5 and 8.

When considering Figure 20, the relatively poorer performance of the *ORG* label becomes more clear, where the strict F1 scores revolve around the 30-35%. After looking into this particular label further, it seems that the models have much trouble identifying the range of these particular entities. For instance, if the full entity text is equal to "Pieter & de Biet Notarissen", then it will recognise "Pieter &" as an entity, while leaving out the rest. Another possibility is that it recognises "Pieter &" and "de Biet Notarissen" as separate entities, which lowers the strict score. Hence, this could explain the relatively large difference between the strict and relaxed scores. Other than this, it also misses quite a few names. As was previously explained, this can be caused by the original pre-training of the *ORG* label. Perhaps it misses many of these entities or predicts the wrong entities as an organisation due to being trained on different datasets previously, while not recognising the financial organisations or formats. It could also be caused by the fact that "Stater" as a company name occurs too often in the training set, which could cause catastrophic forgetting for the other company names and their formatting. Other than this label, the rest of the entities seem to be performing quite well, including the *MISC* label, which is surprising given the low previous scores of this label. This could be caused by the lower number of occurrences of different *MISC* labelled entities within these sets. Another reason for this is that the more general *MISC* entities were easier to predict.

Table 8: The results for the models when considering unique sets of documents or entities.

Model	**Spacy Regex**		**BERTje Regex CRF**		**RobBERT Regex CRF**	
New Set Type	**Documents**	**Entities**	**Documents**	**Entities**	**Documents**	**Entities**
Strict Precision	**80.05%**	83.45%	76.33%	75.30%	74.18%	73.62%
Strict Recall	77.07%	78.37%	**83.39%**	83.05%	78.55%	80.77%
Strict F1	78.53%	80.83%	**79.71%**	78.99%	76.30%	77.03%
Rel Precision	**88.15%**	91.14%	85.36%	88.45%	86.91%	88.13%
Rel Recall	84.81%	84.43%	**91.70%**	92.10%	90.74%	91.88%
Rel F1	84.45%	87.65%	88.42%	90.24%	**88.79%**	89.97%
Strict Sentences	76.56%	72.86%	**83.15%**	75.47%	77.24%	74.01%
Rel Sentences	82.81%	79.24%	**84.10%**	85.54%	83.15%	85.22%

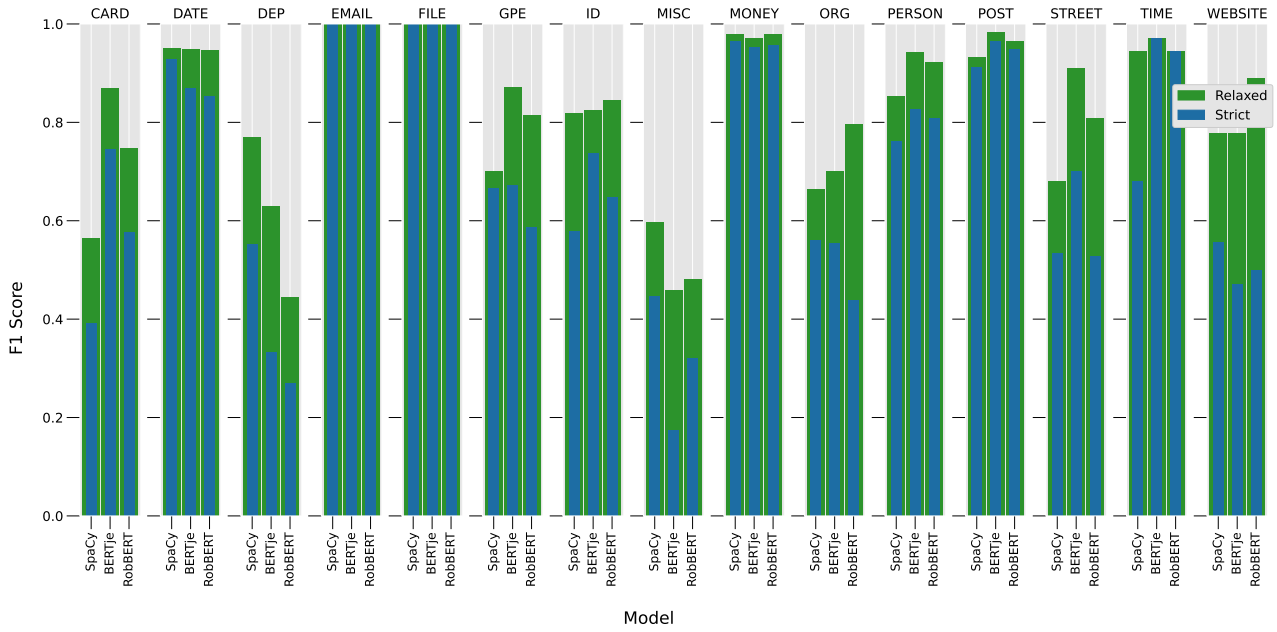

Figure 19: The F1 scores obtained per label and model when only testing on the new documents within the test set.

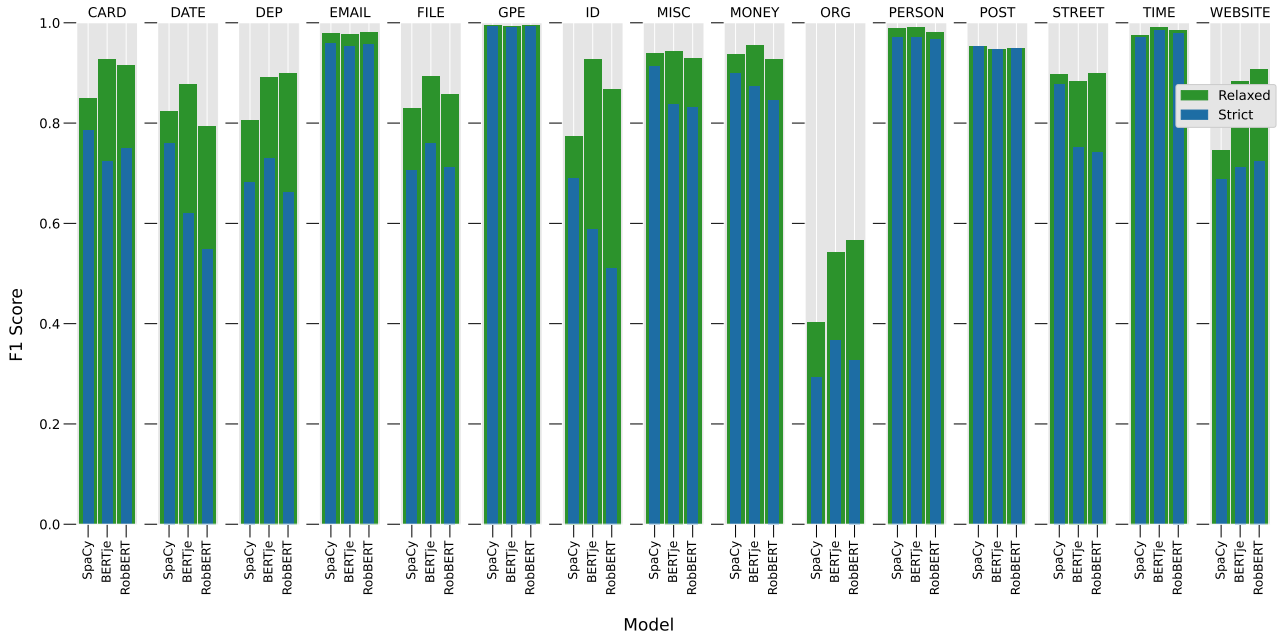

Figure 20: The F1 scores obtained per label and model when only testing on the new entities within the test set.

9 Discussion

This section will discuss various aspects of the methodology used in this thesis. In addition, this section will answer and discuss some of the sub-questions that were posed in the Introduction section. The conclusion of this research and its most prevalent improvements will be discussed in the Conclusion and Future Work section

9.1 Literature and Annotation

First, given current literature, there is little evidence to support the concept of anonymising privacy-based entities in text with the state-of-the-art NER techniques. Most papers are critical of using only NER as a method for anonymising documents [34]. Moreover, some claim that even anonymised documents do not ensure that data is private [32]. The legal experts at Stater also doubt that it is theoretically possible to do so. However, given all the papers shown and discussed in this thesis, there seems to be a general consensus that this technique is crucial for the task of AI-based anonymisation [34, 35]. Additionally, it is also shown that such methods are able to outperform the manual anonymisation of documents [37, 24, 38]. Hence, more research into the topic is always encouraged as this is far from a solved task. Although it is extremely difficult to theoretically proof that this task is doable using only computer programs and AI-based models, these techniques can at the very least assist companies in this task given the problems that big data has introduced [18].

Another point of discussion is the label scheme that was created for this project. Due to the lack of categories in the original label schemes, a new scheme was created that tries to better adhere to the various privacy guidelines provided by the GDPR and by Stater. Although it adheres to most of these guidelines, some topics or terms will not be recognised by the algorithm, such as medical terms, usernames, passwords, political/religious topics and philosophical beliefs. However, even if these labels were included, the model would not be able to learn these labels given the current financial datasets. Therefore, solutions would need to be created for each of the missing labels to fully adhere to the guidelines of both parties. An example would be social media data for political/religious/philosophical topics, synthetic data for usernames/passwords and medical data for medical terms. Additionally, the occurrence of such data would need to be frequent enough in order to prevent phenomena such as "catastrophic forgetting". Another important point of discussion is that fine-grained label schemes were not used, which made certain labels, such as *STREET*, more difficult to predict, given its overlap with the *CARDINAL* label. Hence, it would be interesting to see if such a fine-grained scheme could improve the quality and performance of the models.

The annotation process itself also needs to be discussed. Since there were no clear guidelines for annotating the data beforehand and since there was little time to consider such guidelines, errors were made in both the training and test data. In addition, there was only one annotator for this entire project, which makes the data fairly biased towards certain labelling strategies and detection of certain entity spans. The errors in the test data were mostly fixed with an additional review but there was no time to correct the errors made in the training data. Examples of such errors are the exclusion of money-related words or symbols, such as ("€"/"EUR"), the exclusion of other symbols, such as "<" and "[' ' or misunderstanding the names of organisations, departments or persons. Even though these errors were made, it seems that the models were still able to learn admirably in terms of both the final performance and their generalisability shown in Tables 5 and 8. This conclusion is also supported by the second review of the test set, since this had little influence on the overall outcome (about 0.5% F1 score in general for all models). The reasoning behind the satisfactory learning of the models is most likely due to the large amount of training data that was annotated for this project, which makes these errors not as influential. Therefore, the results shown in this thesis are relatively reliable and rarely influenced by the errors made during the annotation process. The influence of individual biases for the scores and quality of the sets is unknown.

9.2 Models and Results

The choice for these specific NER models can also be debated. Although the SpaCy, BERTje and RobBERT models were the most interesting and favourable models to test for Stater, as is fully discussed in the Model Selection section, there is a high probability that other models can slightly improve upon these results. For instance, it would be interesting to consider the Stanford Dutch NLP model as a replacement for SpaCy to check its performance on the given financial documents. Additionally, the BERT models chosen for this project may not have the best performance overall. However, given the instability of such models, it is unlikely other versions are able to improve upon these models, unless they are specifically trained on the financial domain. Moreover, since these models are managed by the HuggingFace library [119], these models become more manageable in terms of upkeep for the company, which makes them more practical than some of the other versions not maintained by this library.

Another point of contention regarding the models is the stability of the training process for the BERT models. Even if future researchers would experiment with the other Dutch BERT models, it is highly likely that these models would be relatively unstable too in terms of training. This is likely due to the financial domain being

generally more difficult to grasp due to the models not being pre-trained on such domains. The addition of new labels for these models resulted also in more difficult training, which is supported by the scores with and without the Regex system depicted in Table 5. A similar reasoning can be obtained for the CRF model, since its influence on the main results does not seem as substantial as is also observed in Table 5. Therefore, a solution needs to be found where the BERT model can more properly train on more documents within the financial domain. Therefore, these BERT-based models are likely to improve on the strict score if they were fully pre-trained from scratch on a larger and more diverse financial dataset. This could help with both the stability and performance of these models. This trend can be observed for the English pre-trained BERT models, since many variations of this model have been created in recent years, such as a BERT that is based on legal documents [121]. Chalkidis et al. (2020) obtained state-of-the-art performance on the legal domain with this variation on BERT and similar improvements are expected for the financial domain if carried out properly. Another advantage of this is that the difference in training based on the strict or relaxed scores can be assessed, since training on different scores, such as the relaxed score, could improve the performance of the models with respect to the task of de-identification.

Lastly, although the Error Analysis showed that the models were able to generalise relatively well, some labels did underperform a bit compared to the other labels in terms of generalisability. In particular, the labels *ORG*, *MISC* and *STREET* performed noticeably worse than the other labels. There are many causes for the overall lower performance of these labels. First, it was sometimes difficult during the annotation process to set boundaries for the *ORG* and *MISC* entities. Therefore, better boundaries could have been set, which could help the performance of the models, since setting such boundaries was proven to be difficult for the models as well. Second, as was previously noted, a fine-grained label scheme could help both the *ORG* and *STREET* label, since these labels are often confused with *DEP* and *CARD* respectively. However, this would make the training even more difficult than it currently is. Hence, the training would first need to be stabilised better before such schemes are introduced. Third, to help improve the performance of the *MISC* label, the data could be pre-processed better, such that long miscellaneous entities are already removed before the model starts its predictions. However, this would reduce the quality of the de-identified text in general and it means that the text is tampered with. Therefore, it will be difficult to improve the labels given the current rules and dataset provided in this thesis.

10 Conclusion and Future Work

This thesis discusses the task of de-identification using Named Entity Recognition on the financial documents processed by Stater. For this de-identification process, it was first investigated how well previous NER models performed in the literature and their main shortcomings were noted. From the literature, it was concluded that the NER models showed satisfactory performance in general, with an approximate performance of 86% for both Dutch regular expression-based systems and Dutch Deep Learning models. However, one of the shortcomings for such models is the lack of pre-trained models for the Dutch language, which makes it more difficult to properly fine-tune such models on the financial domain. Additionally, it became clear that anonymisation of documents is extremely difficult to prove theoretically and perform practically given the strict requirements for this task. However, it became also clear that NER is a valuable technique that functions as the basis for AI-based de-identification. Although it cannot be guaranteed that documents can be anonymised, it can at least assist in this task. Furthermore, pseudonymisation is a task for which the technique of NER can generally be used given sufficient scores, since it is mostly used as an additional security layer for the original data.

After investigating the literature, a new label scheme was created. This was necessary in order to adhere as much as possible to the guidelines provided in the GDPR and by Stater's privacy documents. Although most guidelines are covered, some categories are left out, due to the lack of specific entities within the provided datasets. With this label scheme, some preprocessing was performed on the given datasets and about 30,000 sentences were annotated using a manually created annotation tool. Next, three different models were tested in total for this thesis on the annotated data. The results from the experiments showed that all three models performed admirably on the data using a combination of regular expressions and deep-learning models. Overall, the SpaCy-based model was able to perform the best on the general "strict score" used throughout the literature. In contrast, the BERT-based models performed better using the manually created "relaxed score", which was specifically made for the task of anonymisation. The generalisability of the models was also tested and these statements hold mostly true for these scores as well. Therefore, if the strict score is important, SpaCy is preferable over the BERT models, while the BERT models are preferable if anonymisation is the relevant task.

Despite the promising results, there are some improvements that will be left to future work. The best possible improvement for this task would be to create a larger and more balanced dataset on which the BERT models can be pre-trained from scratch. Despite their great performance in this thesis, it is possible that, through extensive and stable the training of these BERT models, that these models can outperform the SpaCy model on the strict score in addition to the relaxed score. It is expected that these models, in particular BERTje, are able to learn the vocabulary and tokenisation of this new domain much better. This may also help the CRF model that was used on top the previous models, since this algorithm was not as well trained as it could have been. In addition

to improving the performance of the BERT models, a larger more diverse dataset would be able to include the labels that could not be considered for these experiments. In this way, the models would be able to adhere better to both the GDPR and Stater guidelines for anonymising documents. In addition, clear guidelines will need to be implemented to help with the quality of the data.

Although improving the quality of the data is of the greatest importance, there are other improvements that need to be considered. First, features ought to be added to the BERT models, since a lack of such features most likely resulted in some unstable training and overall inferior results in terms of the strict evaluation metric. Second, a new Dutch SpaCy pre-trained model is bound to release soon at the time of handing in this thesis, since a new transformer pipeline was released for multiple other languages such as English. This is a pipeline based on the Roberta model, which was also used for the RobBERT model within these experiments. Therefore, it will be interesting to observe the differences between this new model and the current results. Another possibility is to combine the label performances of multiple models into one larger model, since the label performance for each model varies much. Furthermore, the de-identification process can be enhanced by adding additional techniques to this process, such as k-anonymisation and differential privacy. Through these techniques, potential individuals can be singled out within these financial documents, which can, in combination with the NER techniques discussed in this project, lead to better de-identification performances. Another method to improve these models is by checking their explainability. Since neither the CRF models nor the SpaCy model can return label probabilities, this was not possible for the current error analysis experiments. However, by manually implementing a CRF model, these probabilities could be returned and, with this knowledge, researchers could gain more insight into the learning of the models and thus further improve the capabilities of the models.

11 Bibliography

[1] Matthew Honnibal, Adriane Boyd, and Vincent D. Warmerdam. Compact word vectors with bloom embeddings. Bloom embeddings blog : https://explosion.ai/blog/bloom-embeddings, April 2022. Image Title: MultiHashEmbed embedding ; Accessed : 10-6-2022.

[2] Matthew Honnibal. Spacy's ner model. https://spacy.io/universe/project/video-spacys-ner-models, April 2022. Accessed: 1-4-2022, Under license of MIT, also founders of the software company Explosion.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[4] Romain Futrzynski. Getting meaning from text: self-attention step-by-step video. Self-attention blog: https://peltarion.com/blog/data-science/self-attention-video, September 2020. Image Title: Self-attention mechanism; Accessed : 16-6-2022.

[5] Usama Khalid, Mirza Omer Beg, and Muhammad Umair Arshad. Rubert: A bilingual roman urdu bert using cross lingual transfer learning. *arXiv preprint arXiv:2102.11278*, 2021.

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

[7] Priyank Jain, Manasi Gyanchandani, and Nilay Khare. Big data privacy: A technological perspective and review. *Journal of Big Data*, 3(1):25, November 2016. ISSN 2196-1115. doi: 10.1186/s40537-016-0059-y.

[8] Roxana Geambasu, Tadayoshi Kohno, Amit A Levy, and Henry M Levy. Vanish: Increasing data privacy with self-destructing data. In *USENIX security symposium*, volume 316, pages 10–5555, 2009.

[9] Lei Xu, Chunxiao Jiang, Jian Wang, Jian Yuan, and Yong Ren. Information Security in Big Data: Privacy and Data Mining. *IEEE Access*, 2:1149–1176, 2014. ISSN 2169-3536. doi: 10.1109/ACCESS.2014.2362522.

[10] Jorge Bernal Bernabe, Jose Luis Canovas, Jose L. Hernandez-Ramos, Rafael Torres Moreno, and Antonio Skarmeta. Privacy-Preserving Solutions for Blockchain: Review and Challenges. *IEEE Access*, 7:164908–164940, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2950872.

[11] European Parlement, the Council, and repealing Directive 95/46/EC. Regulation (eu) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data. http://data.europa.eu/eli/reg/2016/679/oj, 27 April 2016. Access date URL : 11 February 2022 ; Extra note: OJ L 119, 4.5.2016, p. 1–88 (BG, ES, CS, DA, DE, ET, EL, EN, FR, GA, HR, IT, LV, LT, HU, MT, NL, PL, PT, RO, SK, SL, FI, SV).

[12] Sanjay Sharma. *Data Privacy and GDPR Handbook*. John Wiley & Sons, November 2019. ISBN 978-1-119-59424-6.

[13] Sandra Wachter. Normative challenges of identification in the Internet of Things: Privacy, profiling, discrimination, and the GDPR. *Computer Law & Security Review*, 34(3):436–449, June 2018. ISSN 0267-3649. doi: 10.1016/j.clsr.2018.02.002.

[14] Michal S Gal and Oshrit Aviv. The Competitive Effects of the GDPR. *Journal of Competition Law & Economics*, 16(3):349–391, September 2020. ISSN 1744-6414. doi: 10.1093/joclec/nhaa012.

[15] Stater N.V. Mortgage Services — An infosys company. Home page stater. https://stater.nl/en/, 2022. access date: 22-2-2022.

[16] Josep Domingo-Ferrer. Personal Big Data, GDPR and Anonymization. In Alfredo Cuzzocrea, Sergio Greco, Henrik Legind Larsen, Domenico Saccà, Troels Andreasen, and Henning Christiansen, editors, *Flexible Query Answering Systems*, Lecture Notes in Computer Science, pages 7–10, Cham, 2019. Springer International Publishing. ISBN 978-3-030-27629-4. doi: 10.1007/978-3-030-27629-4_2.

[17] Nicholas Vollmer. Recital 26 EU General Data Protection Regulation (EU-GDPR). https://www.privacy-regulation.eu/en/recital-26-GDPR.htm, July 2021.

[18] Junfei Qiu, Qihui Wu, Guoru Ding, Yuhua Xu, and Shuo Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016(1):67, May 2016. ISSN 1687-6180. doi: 10.1186/s13634-016-0355-x.

[19] Philip Russom et al. Big data analytics. *TDWI best practices report, fourth quarter*, 19(4):1–34, 2011.

[20] Dimitrios Kokkinakis and Anders Thurin. Anonymisation of Swedish Clinical Data. In Riccardo Bellazzi, Ameen Abu-Hanna, and Jim Hunter, editors, *Artificial Intelligence in Medicine*, Lecture Notes in Computer Science, pages 237–241, Berlin, Heidelberg, 2007. Springer. ISBN 978-3-540-73599-1. doi: 10.1007/978-3-540-73599-1_31.

[21] Vikas Yadav and Steven Bethard. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. *arXiv:1910.11470 [cs]*, October 2019.

[22] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticæ Investigationes*, 30(1):3–26, January 2007. ISSN 0378-4169, 1569-9927. doi: 10.1075/li.30.1.03nad.

[23] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A Survey on Deep Learning for Named Entity Recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70, January 2020. ISSN 1558-2191. doi: 10.1109/TKDE.2020.2981314.

[24] Vincent Menger, Floor Scheepers, Lisette Maria van Wijk, and Marco Spruit. Deduce: A pattern matching method for automatic de-identification of dutch medical text. *Telematics and Informatics*, 35(4): 727–736, July 2018. ISSN 07365853. doi: 10.1016/j.tele.2017.08.002.

[25] Saul A. Kripke. *Naming and Necessity*. Harvard University Press, 1982.

[26] Xiaoshi Zhong and Erik Cambria. *Time Expression and Named Entity Recognition*, volume 10 of *Socio-Affective Computing*. Springer International Publishing, Cham, 2021. ISBN 978-3-030-78960-2 978-3-030-78961-9. doi: 10.1007/978-3-030-78961-9.

[27] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

[28] AEPD. Agencia española de protección de datos. https://www.aepd.es/es, March 2022. Accessed: 29-3-2022, C/ Jorge Juan, 6. 28001 - Madrid.

[29] The Spanish Data Protection Agency. 10 misunderstandings related to anonymisation. https://edps.europa.eu/data-protection/our-work/publications/papers/aepd-edps-joint-paper-10-misunderstandings-related_en, 27 April 2021. Accessed: 28-3-2022.

[30] TISO/TC 215 Health informatics. Iso 25237:2017 health informatics — pseudonymization. https://www.iso.org/obp/ui/#iso:std:iso:25237:ed-1:v1:en, January 2017. Accessed: 28-3-2022.

[31] Directive (EU) 2019/1024 of the European Parliament and of the Council. open data and the re-use of public sector information. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32019L1024, 29 June 2019. Access date URL : 28-3-2022 ; Extra note: OJ L 172, 26.6.2019, p. 56–83 (BG, ES, CS, DA, DE, ET, EL, EN, FR, GA, HR, IT, LV, LT, HU, MT, NL, PL, PT, RO, SK, SL, FI, SV).

[32] Artur Potiguara Carvalho, Edna Dias Canedo, Fernanda Potiguara Carvalho, and Pedro Henrique Potiguara Carvalho. Anonymisation and compliance to protection data: Impacts and challenges into big data. In *ICEIS (1)*, pages 31–41, 2020.

[33] Liudmyla Vasylieva. Anonymisation and pseudonymisation of textual documents. Master's thesis, University of Oslo ; Faculty of Law, 2021.

[34] Gergely Márk Csányi, Dániel Nagy, Renátó Vági, János Pál Vadász, and Tamás Orosz. Challenges and Open Problems of Legal Document Anonymization. *Symmetry*, 13(8):1490, August 2021. ISSN 2073-8994. doi: 10.3390/sym13081490.

[35] Chaïm van Toledo, Friso van Dijk, and Marco Spruit. Dutch Named Entity Recognition and De-Identification Methods for the Human Resource Domain. *International Journal on Natural Language Computing*, 9(6):23–34, December 2020. ISSN 23194111. doi: 10.5121/ijnlc.2020.9602.

[36] Arttu Oksanen, Minna Tamper, Jouni Tuominen, Aki Hietanen, and Eero Hyvönen. Anoppi: A pseudonymization service for finnish court documents. In *JURIX*, pages 251–254, 2019.

[37] Bennett Kleinberg, Maximilian Mozes, Yaloe van der Toolen, and Bruno Verschuere. Netanos - named entity-based text anonymization for open science, Jun 2017. URL osf.io/w9nhb.

[38] Jihang Mao and Wanli Liu. Hadoken: a bert-crf model for medical document anonymization. In *IberLEF@ SEPLN*, pages 720–726, 2019.

[39] Miranda Mourby, Elaine Mackey, Mark Elliot, Heather Gowans, Susan E. Wallace, Jessica Bell, Hannah Smith, Stergios Aidinlis, and Jane Kaye. Are 'pseudonymised' data always personal data? Implications of the GDPR for administrative data research in the UK. *Computer Law & Security Review*, 34(2): 222–233, April 2018. ISSN 02673649. doi: 10.1016/j.clsr.2018.01.002.

[40] Chaïm van Toledo and Marco R Spruit. Adopting privacy regulations in a data warehouse-a case of the anonimity versus utility dilemma. In *KDIR*, pages 234–239, 2016.

[41] Hanna Berg, Aron Henriksson, and Hercules Dalianis. The Impact of De-identification on Downstream Named Entity Recognition in Clinical Text. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 1–11, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.louhi-1.1.

[42] Rahul Sharnagat. Named entity recognition: A literature survey. *Center For Indian Language Technology*, pages 1–27, 2014.

[43] Ralph Grishman and Beth Sundheim. Message Understanding Conference- 6: A Brief History. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996.

[44] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *arXiv:cs/0306050*, June 2003.

[45] 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 *Lrec*, volume 2, pages 837–840. Lisbon, 2004.

[46] Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. Overview of the TREC 2010 Entity Track. Technical report, NORWEGIAN UNIV OF SCIENCE AND TECHNOLOGY TRONDHEIM, November 2010.

[47] Diego Mollá, Menno van Zaanen, and Daniel Smith. Named Entity Recognition for Question Answering. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 51–58, Sydney, Australia, November 2006.

[48] Alex Brandsen, Suzan Verberne, Karsten Lambers, and Milco Wansleeben. Can BERT Dig It? – Named Entity Recognition for Information Retrieval in the Archaeology Domain. *arXiv:2106.07742 [cs]*, June 2021.

[49] Veena Gangadharan and Deepa Gupta. Recognizing Named Entities in Agriculture Documents using LDA based Topic Modelling Techniques. *Procedia Computer Science*, 171:1337–1345, January 2020. ISSN 1877-0509. doi: 10.1016/j.procs.2020.04.143.

[50] Mohammad Ebrahim Khademi and Mohammad Fakhredanesh. Persian Automatic Text Summarization Based on Named Entity Recognition. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, July 2020. ISSN 2228-6179, 2364-1827. doi: 10.1007/s40998-020-00352-2.

[51] Bogdan Babych and Anthony Hartley. Improving Machine Translation Quality with Automatic Named Entity Recognition. In *Proceedings of the 7th International EAMT Workshop on MT and Other Language Technology Tools, Improving MT through Other Language Technology Tools, Resource and Tools for Building MT at EACL 2003*, 2003.

[52] Dimitra Farmakiotou, Vangelis Karkaletsis, John Koutsias, George Sigletos, Constantine D. Spyropoulos, and Panagiotis Stamatopoulos. Rule-Based Named Entity Recognition For Greek Financial Texts. In *In Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries (COMLEX 2000*, pages 75–78, 2000.

[53] Isabel Segura-Bedmar, Paloma Martinez, and Maria Herrero Zazo. SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, page 10. Association for Computational Linguistics, 2013.

[54] Alireza Mansouri, Lilly Suriani Affendey, and Ali Mamat. Named entity recognition approaches. *International Journal of Computer Science and Network Security*, 8(2):339–344, 2008.

[55] Ji-Hwan Kim and Philip C Woodland. A rule-based named entity recognition system for speech input. In *Sixth International Conference on Spoken Language Processing*, 2000.

[56] George R. Krupka and Kevin Hausman. IsoQuest inc.: Description of the nerowl extractor system as used for muc-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*, 1998. URL https://aclanthology.org/M98-1015.

[57] Andrei Mikheev, Marc Moens, and Claire Grover. Named Entity Recognition without Gazetteers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 1–8, Bergen, Norway, June 1999. Association for Computational Linguistics.

[58] Enrique Alfonseca and Suresh Manandhar. An unsupervised method for general named entity recognition and automated concept discovery. In *Proceedings of the 1st international conference on general WordNet, Mysore, India*, pages 34–43, 2002.

[59] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*, 1992.

[60] Yusuke Shinyama and Satoshi Sekine. Named Entity Discovery Using Comparable News Articles. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 848–853, Geneva, Switzerland, August 2004. COLING.

[61] Philipp Cimiano and Johanna Völker. Towards large-scale, open-domain and ontology-based named entity classification. In *Proceedings of RANLP*, volume 5, pages 166–172, 2005.

[62] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence*, 165(1):91–134, June 2005. ISSN 0004-3702. doi: 10.1016/j.artint.2005.03.001.

[63] David Nadeau, Peter D. Turney, and Stan Matwin. Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity. In Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Ahmed Y. Tawfik, and Scott D. Goodwin, editors, *Advances in Artificial Intelligence*, volume 3060, pages 266–277. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-22004-6 978-3-540-24840-8. doi: 10.1007/11766247_23.

[64] Shaodian Zhang and Noémie Elhadad. Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts. *Journal of Biomedical Informatics*, 46(6):1088–1098, December 2013. ISSN 1532-0464. doi: 10.1016/j.jbi.2013.08.004.

[65] Michael Collins and Yoram Singer. Unsupervised Models for Named Entity Classification. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[66] Maksim Tkachenko and Andrey Simanovsky. Named entity recognition: Exploring features. In *KONVENS*, pages 118–127, 2012.

[67] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: A High-Performance Learning Name-finder. *arXiv:cmp-lg/9803003*, March 1998.

[68] Shihong Yu, Shuanhu Bai, and Paul Wu. Description of the Kent Ridge Digital Labs System Used for MUC-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998*, 1998.

[69] Michael Collins. Ranking Algorithms for Named Entity Extraction: Boosting and the VotedPerceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073165.

[70] Jason P.C. Chiu and Eric Nichols. Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370, July 2016. ISSN 2307-387X. doi: 10.1162/tacl_a_00104.

[71] Andrei Mikheev. A Knowledge-free Method for Capitalized Word Disambiguation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 159–166, College Park, Maryland, USA, June 1999. Association for Computational Linguistics. doi: 10.3115/1034678.1034710.

[72] David McDonald. Internal and External Evidence in the Identification and Semantic Categorization of Proper Names. In *Acquisition of Lexical Knowledge from Text*, 1993.

[73] R. Gaizauskas, T. Wakao, K. Humphreys, H. Cunningham, and Y. Wilks. University of Sheffield: Description of the LaSIE System as Used for MUC-6. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*, 1995.

[74] Sam Coates-Stephens. The Analysis and Acquisition of Proper Names for the Understanding of Free Text. *Computers and the Humanities*, 26(5):441–456, December 1992. ISSN 1572-8412. doi: 10.1007/BF00136985.

[75] Yoshimasa Tsuruoka and Jun'ichi Tsujii. Boosting Precision and Recall of Dictionary-Based Protein Name Recognition. In *Proc. of the ACL-03 Workshop on Natural Language Processing in Biomedicine*, pages 41–48. Citeseer, 2003.

[76] Hema Raghavan and James Allan. Using Soundex Codes for Indexing Names in ASR Documents. In *Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL 2004*, pages 22–27, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[77] Christine Thielen. An Approach to Proper Name Tagging for German. *arXiv:cmp-lg/9506024*, June 1995.

[78] Jianhan Zhu, Victoria Uren, and Enrico Motta. ESpotter: Adaptive Named Entity Recognition for Web Browsing. In Klaus-Dieter Althoff, Andreas Dengel, Ralph Bergmann, Markus Nick, and Thomas Roth-Berghofer, editors, *Professional Knowledge Management*, Lecture Notes in Computer Science, pages 518–529, Berlin, Heidelberg, 2005. Springer, Springer. ISBN 978-3-540-31620-6. doi: 10.1007/11590019_59.

[79] Thierry Poibeau. Dealing with Metonymic Readings of Named Entities. *arXiv:cs/0607052*, July 2006.

[80] Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. SVM Based Learning System for Information Extraction. In Joab Winkler, Mahesan Niranjan, and Neil Lawrence, editors, *Deterministic and Statistical Methods in Machine Learning*, Lecture Notes in Computer Science, pages 319–339, Berlin, Heidelberg, 2005. Springer. ISBN 978-3-540-31728-9. doi: 10.1007/11559887_19.

[81] Rodrigo Agerri and German Rigau. Robust multilingual Named Entity Recognition with shallow semi-supervised features. *Artificial Intelligence*, 238:63–82, September 2016. ISSN 0004-3702. doi: 10.1016/j.artint.2016.05.003.

[82] Lin Yao, Hong Liu, Yi Liu, Xinxin Li, and Muhammad Waqas Anwar. Biomedical Named Entity Recognition based on Deep Neutral Network. *International Journal of Hybrid Information Technology*, 8(8):279–288, 2015. ISSN 17389968. doi: 10.14257/ijhit.2015.8.8.29.

[83] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, September 2013.

[84] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

[85] Changhan Wang, Kyunghyun Cho, and Douwe Kiela. Code-Switched Named Entity Recognition with Embedding Attention. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 154–158. Association for Computational Linguistics, 2018. doi: 10.18653/v1/W18-3221.

[86] Onur Kuru, Ozan Arkan Can, and Deniz Yuret. CharNER: Character-Level Named Entity Recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 911–921. The COLING 2016 Organizing Committee, 2016.

[87] Bill Y. Lin, Frank Xu, Zhiyi Luo, and Kenny Zhu. Multi-channel BiLSTM-CRF Model for Emerging Named Entity Recognition in Social Media. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 160–165. Association for Computational Linguistics, 2017. doi: 10.18653/v1/W17-4421.

[88] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[89] Qikang Wei, Tao Chen, Ruifeng Xu, Yulan He, and Lin Gui. Disease named entity recognition by combining conditional random fields and bidirectional recurrent neural networks. *Database*, 2016, 10 2016. ISSN 1758-0463. doi: 10.1093/database/baw140. URL https://doi.org/10.1093/database/baw140. baw140.

[90] Vikas Yadav, Rebecca Sharp, and Steven Bethard. Deep Affix Features Improve Neural Named Entity Recognizers. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 167–172, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/S18-2021.

[91] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[92] 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(ARTICLE):2493–2537, 2011.

[93] Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. Fast and accurate entity recognition with iterated dilated convolutions. *arXiv preprint arXiv:1702.02098*, 2017.

[94] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[95] Nut Limsopatham and Nigel Collier. Bidirectional LSTM for Named Entity Recognition in Twitter Messages. *Proceedings of the 2nd Workshop on Noisy User-generated Text*, 2016. doi: 10.17863/CAM.7201.

[96] Andrej Žukov Gregorič, Yoram Bachrach, and Sam Coope. Named Entity Recognition With Parallel Recurrent Neural Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 69–74. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-2012.

[97] Peng-Hsuan Li, Ruo-Ping Dong, Yu-Siang Wang, Ju-Chieh Chou, and Wei-Yun Ma. Leveraging Linguistic Structures for Named Entity Recognition with Bidirectional Recursive Neural Networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2664–2669. Association for Computational Linguistics, 2017. doi: 10.18653/v1/D17-1282.

[98] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*, 2017.

[99] Matthew E. 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. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-1202.

[100] Fien De Meulder, Walter Daelemans, and Véronique Hoste. *A Named Entity Recognition System for Dutch*. Brill, January 2002. ISBN 978-90-04-33403-8. doi: 10.1163/9789004334038_008.

[101] William W. Cohen. Fast Effective Rule Induction. In Armand Prieditis and Stuart Russell, editors, *Machine Learning Proceedings 1995*, pages 115–123. Morgan Kaufmann, San Francisco (CA), January 1995. ISBN 978-1-55860-377-6. doi: 10.1016/B978-1-55860-377-6.50023-2.

[102] Xavier Carreras, Lluís Màrquez, and Lluís Padró. Named Entity Extraction using AdaBoost. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[103] Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. BERTje: A Dutch BERT Model. *arXiv preprint arXiv:1912.09582*, December 2019. doi: 10.48550/arXiv.1912.09582.

[104] Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. Improving low resource named entity recognition using cross-lingual knowledge transfer. In *IJCAI*, volume 1, pages 4071–4077, 2018.

[105] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[106] Pieter Delobelle, Thomas Winters, and Bettina Berendt. Robbert: a dutch roberta-based language model. *arXiv preprint arXiv:2001.06286*, 2020.

[107] Mikhail Y Arkhipov, Mikhail S Burtsev, et al. Application of a hybrid bi-lstm-crf model to the task of russian named entity recognition. In *Conference on artificial intelligence and natural language*, pages 91–103. Springer, 2017.

[108] Anu Thomas and S Sangeetha. Performance analysis of the state-of-the-art neural named entity recognition model on judicial domain. In *Soft Computing: Theories and Applications*, pages 147–154. Springer, 2020.

[109] Shotaro Misawa, Motoki Taniguchi, Yasuhide Miura, and Tomoko Ohkuma. Character-based bidirectional lstm-crf with words and characters for japanese named entity recognition. In *Proceedings of the first workshop on subword and character level models in NLP*, pages 97–102, 2017.

[110] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*, 2021.

[111] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

[112] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

[113] Roeland Ordelman, Franciska de Jong, Arjan Van Hessen, and Hendri Hondorp. Twnc: a multifaceted dutch news corpus. *ELRA Newsletter*, 12(3/4):4–7, 2007.

[114] Nelleke Oostdijk, Martin Reynaert, Véronique Hoste, and Ineke Schuurman. The construction of a 500-million-word reference corpus of contemporary written dutch. In *Essential speech and language technology for Dutch*, pages 219–247. Springer, Berlin, Heidelberg, 2013.

[115] 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. *arXiv preprint arXiv:1907.11692*, 2019.

[116] Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache, 2019.

[117] Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. Portuguese named entity recognition using bert-crf. *arXiv preprint arXiv:1909.10649*, 2019.

[118] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[119] Hugging Face. Hugging face, the ai community building the future. Hugging Face information : https://huggingface.co/huggingface, 2022. Last Accessed : 24-7-2022.

[120] Kemal Kurniawan. pytorch-crf. pytorch-crf documentation : https://pytorch-crf.readthedocs.io/en/stable/#, 2019. adjustment of the original by [122] ; Accessed : 10-6-2022.

[121] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*, 2020.

[122] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[123] GR Krupka and K IsoQuest. Description of the nerowl extractor system as used for muc-7. In *Proc. 7th Message Understanding Conf*, pages 21–28, 2005.

[124] Richard Evans and Stafford Street. A framework for named entity recognition in the open domain. *Recent advances in natural language processing III: selected papers from RANLP*, 260(267-274):110, 2003.

[125] GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th annual meeting of the association for computational linguistics*, pages 473–480, 2002.

[126] Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. Svm based learning system for information extraction. In *International Workshop on Deterministic and Statistical Methods in Machine Learning*, pages 319–339. Springer, 2004.

[127] Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. Cloze-driven pre-training of self-attention networks. *arXiv preprint arXiv:1903.07785*, 2019.

[128] Explosion AI. Spacy dutch models. Dutch model links : https://spacy.io/models/nl, 2022. Last Accessed : 27-07-2022.

12 Appendix

Literature

Table 9: An overview of the NER results of multiple papers from the literature study. *multiple versions, datasets or scores were found for the given model, hence the average of these scores were calculated for simplicity. The full results can be found in the original papers.

Paper	Method type	Language	Dataset	F1 Score
Farmakiotou et al. (2000)	Rule/Pattern-based	Greek	Kapa-TEL dataset	*83,63%
Krupka and IsoQuest (2005)	Rule/Pattern-based	English	MUC-7	*85,39%
Menger et al. (2018)	Rule/Pattern-based	Dutch	UMC test corpus	86.20%
Collins and Singer (1999)	Unsupervised	English	MUC-6	83.1%
Evans and Street (2003)	Unsupervised	English	Custom	92.25%
Carreras et al. (2002)	Supervised	Dutch	CoNLL02	77,05%
Zhou and Su (2002)	Supervised	English	MUC-7	94.1%
Li et al. (2004)	Supervised	English	CoNLL03	88.3%
Baevski et al. (2019)	Deep Learning	English	CoNLL 2003	93.5%
AI (2022)	Deep Learning	Dutch	*LassySmall/Alpino/other	77.25%
de Vries et al. (2019)	BERT-based	Dutch	*CoNLL/SoNaR/SpaCy	$\approx$ 87.09%
Delobelle et al. (2020)	*BERT-based	Dutch	CoNLL02	$\approx$ 88,31%

Data Exploration

Table 10: Tested probability distributions on the file size data. The standard threshold value of p=0.05 is used for testing whether the data follows a certain distribution.

Distribution	p-value	Rejected?
Normal	1.028^{-14}	yes
Lognormal	2.619^{-13}	yes
Weibull Minimum Extreme Value	2.272^{-11}	yes
Weibull Maximum Extreme Value	5.682^{-70}	yes
General Extreme Value	0.004	yes
Rayleigh	3.601^{-20}	yes
Moyal	1.168^{-18}	yes
Gamma	2.276^{-18}	yes
Fisk	0.003	yes

Table 11: All the regular expression (regex) types used throughout the document selection process.

Regex type	**Re-used**	**Description and Reasoning**
Email headers	Yes	Headers occur too frequently in all emails. To avoid bias towards learing these headers, these should not be included too often in the training data. Therefore, they should be excluded from the email strings during selection.
Email templates	Yes	Some templates by companies include certain lines of texts, such as "Heeft u vragen over deze e-mail" or "Dit bericht en alle eventuele attachments". Since these also occur too frequently in comparison to the dynamic content of the emails, they also are excluded from these selections.
Email ending phrases	Yes	Most emails close off with some form of final addressal, such as "Met vriendelijke groeten". Since these do not include any entities generally speaking, unlike the beginnings of the emails, they will not be included in the document selection process.
Indexation	No	Since indexations can easily be handled outside of the NER systems using regular expressions, these will not be included in the selection process.
Disclaimers and Statements	Partially	There will be many emails where disclaimers appear on the bottom of these documents. Although these disclaimers can contain useful entities, and should therefore be included occasionally, too many of them will result in the models trying to detect exactly this text and not generalise to other text. Therefore, these entire sections ought to be excluded. This includes any English versions of these disclaimers or statements.
Links	No	There are many links in these documents that account for a large portion of the emails. This is due to most links being straight copies of the internet address. As a result, they should be excluded from the document selection. This also includes long emails that do not occur in the header of these emails.
Attachments	No	A similar reasoning can be applied to attachments when compared to links. Although most of these long names occur in the email header "Attachments:", they sometimes also appear in other parts of the text. Hence, they will also be removed.
Whitespace-related	Partially	Since these emails can be considered as unstructured text, it occurs regularly that multiple breaklines, tabs or whitespaces are written in these texts. Since they can account for the majority of an emails size, they will be excluded and either replaced by a single whitespace or simply removed, depending on the number of such patterns occurring in the text.

Table 12: Statistics of the unique inquiry directories chosen for the annotation process.

Document Type	# Files	% of Total	Data type
“Aanneemovereenkomst”	6	00.07%	text
“Afschrift Betaalrekening”	2850	34.90%	table
“Bouw specificatie offerte”	6	00.07%	table
“Details bijschrijving”	69	00.84%	table
“Company a”	34	00.41%	hybrid
“Eigenaarsinformatie”	7	00.08%	hybrid
“Jaaroverzicht”	125	01.53%	hybrid
“Loonstrook”	3295	40,35%	table
“Notaris stuk”	7	00.08%	hybrid
“Offerte iets”	38	00.47%	hybrid
“Pensioen overzicht”	896	10.97%	hybrid
“Salarisspecificatie loonstrook”	9	00.11%	table
“Schenkingsovereenkomst”	23	00.28%	text
“Taxatie”	83	01,01%	text
“Taxatie Company b”	720	08,82%	text
Total	8167	100%	-

Annotation Process

Stater definitions de-identification techniques

- Verwerking persoonsgegevens: “alle handelingen die gedaan kunnen worden met persoonsgegevens.”
- De-personaliseren: “vorm van generaliseren in het kader van anonimiseren”.
- Pseudonimiseren: “procedure waarmee identificerende gegevens met een bepaald algoritme worden vervangen door versleutelde gegevens (het pseudoniem). Het algoritme kan voor een persoon altijd hetzelfde pseudoniem berekenen, waardoor informatie over de persoon, ook uit verschillende bronnen, kan worden gecombineerd. ”
- Maskeren: “vervangen van (een deel van de) data in een veld door een bepaald teken, zodat het risico op misbruik verminderd wordt.”
- Anonimiseren: procedure waarmee identificerende gegevens worden verwijderd of vervangen, waarbij het koppelen op persoon van informatie uit verschillende bronnen niet mogelijk is.”
- Herleidbaarheid: “de mogelijkheid om een persoon in de dataset te individualiseren door sommige of alle records uit te licht.”
- Koppelbaarheid: “zijnde de mogelijkheid om ten minste twee records over deze betrokkene of groep met elkaar in verband te brengen (correlatie op grond van een combinatie van ten minste twee records) met een andere gegevensverzameling.”
- Deduceerbaarheid: de mogelijkheid om de waarde van een persoonskenmerk met grote waarschijnlijkheid af te leiden.”

Annotation tool actions

- Move action: move to the next token, which implies that the token is not a named entity.
- Annotate token action: if the token is a named entity, enter the annotation phase, so that the token can be annotated.
- Enter label action: after writing down the label, this will mark the token accordingly and continue to the next token
- Restart sentence: If the annotator made an error, they can restart the current line. Going to the previous token causes many correlation issues between the tokens and therefore, the user does need to restart the entire line.

- Skip sentence: If the sentence does not contain any named entities, the user can skip this entire sentence and move on to the next sentence, if there is one.
- Previous sentence: if an error was made in the previous sentence, the user can go back one sentence.
- End current entity span: if the entity span contains two different tokens of the same type, the user needs to end the entity span directly, which is enabled through this action.

Table 13: A table containing all of the regular expressions used throughout the annotation process.

Regex type	**Regex syntax**	**Regex semantics**
Weekdays	`([mM]aandag|[Dd]insdag|[Ww]oensdag|[Dd]onderdag|[Vv]rijdag|[Zz]aterdag|[Zz]ondag)`	All days of the week, both with and without capital letters.
Months	`([Jj]anuari|[Ff]ebruari|[Mm]aart|[Aa]pril|[Mm]ei|[Jj]uni|[Jj]uli|[Aa]ugustus|[Ss]eptember|[Oo]ktober|[Nn]ovember|[Dd]ecember)`	All months of the year in word format with and without capital letters.
Month numbers	`(12|11|10|0\d{1}|\d{1})`	Months in number format.
Day numbers	`(31|30|2\d{1}|1\d{1}|0\d{1}|\d{1})`	All possible days in the month.
Years	`(1\d{3}|2\d{3})`	All year numbers from year 1000 to 2999.
Hours	`(24|23|22|21|20|1\d{1}|0\d{1}|\d{1})`	All hours in a day
Minutes	`(60|5\d{1}|4\d{1}|3\d{1}|2\d{1}|1\d{1}|0\d{1})`	All minutes in an hour. Can also be used for seconds, although the format for seconds changes frequently.
File formats	`(png|pdf|jpg|jpeg|csv|xlsx|txt|rdf|msg|gif)`	All file formats encountered or often used throughout both datasets.
First and surname pattern	`"(" + "|".join(str(name) for name in name_list) + ")"`	A pattern for combining multiple names in the Name-based patterns.
Email 1	`(\[|<|')(mailto:)?[._\w-]+@[._\w-]+(\.com|\.nl|\.org)(\]|>|')((\]|>|'))?`	Finds the email containing including open or closed brackets
Email 2	`[._\w-]+@[._\w-]+(\.com|\.nl|\.org/`	Finds the emails without the brackets
Website	`(\[|<|\{)*(http(s)?://)?(www\.)?[\w\\/:.-]+(\.com|\.nl|\.org|\.tv|\.ly)[\w\\/:?%=_&.-]*(\]|>|\})*`	A pattern to find anything related to links on the World Wide Web.
Date 1	`(^|(?<=[^\w]))((" + years + "[-\s]" + month numbers + "[-\s]" + day numbers + ")|(" + day numbers +"[-\s](" + month numbers + "|" + months + ")[-\s]" + + ")|"(" + week days + "( |, )" + day numbers + "\.? " + months + " " + years + "))(\$|(?=[^\w]))`	All dates that include a year, month and day format.

continues on next page

Table 13: A table containing all of the regular expressions used throughout the annotation process.

Regex type	Regex syntax	Regex semantics
Date 2	`(^|(?<=[^\w]))(" + day numbers + "[-\s]" + months + " " + ")|(" + months + "[-\s]" + years + ")(\$|(?=[^\w]))`	Dates only including either weekday and month or month and year.
Date 3	`(^|(?<=[^\w]))(" + week days + "|" + months + ")|(" + years + ")(\$|(?=[^\w]))`	Individual date formats for weekdays, months or years.
Time 1	`(^|(?<=[^\w]))((" + hours + "[:.]{1}"+ minutes + "[:.]\d{2})|(" + hours + "[:.]" + minutes + " hours)) (\$|(?=[^\w]))`	Any time format that includes hours, minutes and seconds or a format where the word "uur" is included
Time 2	`("+hours+":"+minutes+")"`	A less strict version of "Time 1" pattern, which should only be used after the previous patterns.
File 1	`(?<=[:,] ) [\+\\)\\(\s\w\\/:?%=_&.-]+\." + file formats + "(\$|(?=[^\w]))`	File format pattern for the email header.
File 2	`(^|(?<=[^\w])) [\+\w\\/:?%=_&.-]+\." + file_formats + "(\$|(?=[^\w]))`	File format pattern for all other cases
Postcodes	`(^|(?<=[^\w]))((\d{4}(|-|)([A-Z]|[a-z]){2})|([Pp]ostbus( |-|)\d{1,5})) (\$|(?=[^\w]))`	A pattern to detect both postcodes and post office boxes.
Money	`(^|(?<=[^\w]))(€|EUR)( |) (\d{1,3})((\.\d{3})+)? ((,\d{2})?)( EUR)? (\$|(?=[^\w]))`	A pattern to guaranteedly finds money-based entities in text.
Lexicon pattern	`(^|(?<=[^\w])) (" + "|".join(str(word) for word in lexicon) + ")(\$|(?=[^\w]))`	A pattern that can be applied to any lexicon type data, so that all names within the given list can be recognised. This includes the following lexicons: *GPE* and *ID*
Full name	`"(^|(?<=[^\w]))(" + firstname_pattern + "(, | ))+" + surname_pattern + "((-| -| )" + surname_pattern + ")? (\$|(?=[^\w]))"`	Check for a full name, where the first name can occur multiple times and a person can have multiple surnames.
Abbreviated name	`"(^|(?<=[^\w])) (([A-Z]\.)+(, | )" + surname_pattern + ")((-| - | )" + surname_pattern + ")? (\$|(?=[^\w]))"`	Used to find abbreviated names and, if applicable, multiple surnames, which are generally detected by people who are married.
Full name reversed	`"(^|(?<=[^\w]))(" + achternamen_pattern + "((-| -| )+ achternamen_pattern + ")? (, | ))(" + voornamen_pattern + (\$|(?=[^\w]))"`	Same as the "Full name" pattern, but in reverse with some added delimiters.

Table 14: Post annotation statistics including duplicates

Category	**Email**		**Inquiries**		**Totals**				
Labels	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Emails**	**Inquiries**	**Total**
CARDINAL	715	207	465	236	1180	443	922	701	1623
DATE	934	340	1390	758	2324	1098	1274	2148	3422
DEPARTMENT	1115	356	188	99	1303	455	1471	287	1758
EMAIL	2496	780	27	0	2523	780	3276	27	3303
FILE	838	246	4	15	842	261	1084	19	1103
FUNCTION	248	89	8	3	256	92	337	11	348
GPE	869	247	440	207	1309	454	1116	647	1763
ID	995	339	843	401	1838	740	1334	1244	2578
MISC	257	96	268	159	525	255	353	427	780
MONEY	208	94	1913	1200	2121	1294	302	3113	3415
NORP	4	0	7	3	11	3	4	10	14
ORGANISATION	2378	657	509	290	2887	947	3035	799	3834
PERSON	1625	548	265	209	1890	757	2173	474	2647
POSTCODE	711	222	192	133	903	355	933	325	1258
STREET	473	142	204	111	677	253	615	315	930
TIME	746	254	234	28	980	282	1000	262	1262
WEBSITE	812	284	108	78	920	362	1096	186	1282
Total	15424	4901	7065	3930	22489	8831	20325	10995	31320

Table 15: Post annotation statistics including duplicates

Category	**Email**		**Inquiries**		**Totals**				
Labels	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Emails**	**Inquiries**	**Total**
CARDINAL	715	207	465	236	1180	443	922	701	1623
DATE	934	340	1390	758	2324	1098	1274	2148	3422
DEPARTMENT	1115	356	188	99	1303	455	1471	287	1758
EMAIL	2496	780	27	0	2523	780	3276	27	3303
FILE	838	246	4	15	842	261	1084	19	1103
FUNCTION	248	89	8	3	256	92	337	11	348
GPE	869	247	440	207	1309	454	1116	647	1763
ID	995	339	843	401	1838	740	1334	1244	2578
MISC	257	96	268	159	525	255	353	427	780
MONEY	208	94	1913	1200	2121	1294	302	3113	3415
NORP	4	0	7	3	11	3	4	10	14
ORGANISATION	2378	657	509	290	2887	947	3035	799	3834
PERSON	1625	548	265	209	1890	757	2173	474	2647
POSTCODE	711	222	192	133	903	355	933	325	1258
STREET	473	142	204	111	677	253	615	315	930
TIME	746	254	234	28	980	282	1000	262	1262
WEBSITE	812	284	108	78	920	362	1096	186	1282
Total	15424	4901	7065	3930	22489	8831	20325	10995	31320

Models

Table 16: The architecture of BERT base and BERT large according to Devlin et al. (2018).

Description	**BERT base**	**BERT large**
Transformer blocks	12	24
Hidden size	768	1024
Attention heads	12	16
Total parameters	110 million	340 million

Setup

Table 17: The relevant hyperparameters of the models used for the experiments.

	SpaCy Models	BERT base + regex	multicolumn1cBERT CRF
Number of Epochs	10	4	4
Optimizer	Adam	AdamW	AdamW
Learning rate	5e-3	3e-5	5e-5
Beta 1	0.9	0.9	0.9
Beta 2	0..999	0.999	0.999
Epsilon	1e-8	1e-8	1e-8
L2	1e-6	None	None
Learning Rate Scheduler	Yes (manual)	No	No
step_size	0.01/(i+1)*2)	None	None

Table 18: A table containing all of the regular expressions used throughout the annotation process.

Regex type	**Regex syntax**	**Regex semantics**
File formats	`(png|pdf|jpg|jpeg|csv|xlsx|txt|rdf|msg|gif|bin)`	All file formats encountered or often used throughout both datasets.
Email 1	`([|< |<|')?([|<|')(mailto:)?[._\w-]+@[._\w-]+(\.com|\.nl|\.org|\.de|\.info|\.nu|\.lan|\.eu)(]|>|')((]|>|'))?`	Finds the email containing including open or closed brackets
Email 2	`([|<|<|'|()?(mailto:)?[._\w-]+@[._\w-]+(\.com|\.nl|\.org|\.de|\.info|\.nu|\.lan|\.eu)(]|>|'|))?`	Finds the emails without the brackets
Website 1	`([|<|\{)*(http(s)?://)?(www\.)?[\w\\/:.-]+(\.com|\.nl|\.org|\.tv|\.ly|\.nu|\.me|\.aspx|\.de|\.info|\.nu|\.lan|\.eu))[\w\\/\+#:?%=_&.-]*(]|>|\})*`	A pattern to find anything related to links on the World Wide Web that includes some form of extension.
Website 2	`([|<|\{)+((http(s)?://)|(www\.))[\w\\/\+#:?%=_&.-]*(]|>|})+`	A pattern to find any extra websites that contains at least the HTTP or World Wide Web extension and brackets.
File 1	`(?<=[:,] )[\+\\)\\(\s\w\\/:?%=_&.-]+" + file formats + "(\$|(?=[^\w]))`	File format pattern for the email header.
File 2	`(^|(?<=[^\w]))[\+\w\\/:?\%=_&.-]+" + file_formats + "(\$|(?=[^\w]))`	File format pattern for all other cases, where we cannot filter on whitespace-based characters
Postcodes	`(^|(?<=[^\w]))((\d{4}(|-|)([A-Z]|[a-z]){2}|([Pp]ostbus( |-|)\d{1,5}))(\$|(?=[^\w]))`	A pattern to detect both postcodes and post office boxes.

continues on next page

Table 18: A table containing all of the regular expressions used throughout the annotation process.

Regex type	Regex syntax	Regex semantics
Postcode Exceptions	`["ad","af","al","bv","cd","dj", "en","id","is","in""ja","na","n u","of","op","om","pa","te""vs" ,"za","zo"]`	All the words that will not be considered for postcodes if they are lower-cased.
Money	`(^|(?<=[^\w%]))((-|\+) | (-|\+))?(€|EUR|EURO|EUR | EURO )([ \t\n\r\f\v.+-]+|) (\d{1,3})(((\.)?\d{3})+)? (,\d{2}|,--|,-,|, -|,-|)? ($|(?=[^\w%]))(?! %)`	A pattern to finds money-based entities that have a prefix based on the currency "Euro". Note that only Euro-based currencies are searched for explicitly.
Money 2	`(^|(?<=[^\w%]))((-|\+) | (-|\+))?(\d{1,3})(((\.)? \d{3})+)?(,\d{2}( EURO|EURO| EUR|EUR| cent|cent)|(,\d{2}| ,-|, -| EURO|EURO| EUR|EUR| cent|cent))($|(?=[^\w%]))(?! %)`	A pattern to finds money-based entities that have a suffix based on the currency "Euro".
Money 3	`(^|(?<=[^\w%]))( (-|\+)| (-|\+) |(-|\+)|)\d{1,3}(((\.|,) \d{3})+)?((,|\.)\d{2}|(,|\.) -|(,|\.) -)($|(?=[^\w%]))(?! %)`	A pattern to finds money-based entities that have some form of decimal indication, such as ",-".
Money 4	`(^|(?<=[^\w%]))( (-|\+)| (-|\+) |(-|\+)|)\d{1,3}(,\d{3})+ (\.\d{2}|(,|\.)-|(,|\.) -)? ($|(?=[^\w%]))(?! %)`	A pattern to find all potential money-based entities. Note: this will not overwrite the label "CARDINAL".
Unique Text Money	`["een","n","twee","drie","vier" ,"vijf","zes","zeven","acht","n egen","tien","elf","twaalf","de rtien","viertien","vijftien","z estien","zeventien","achttien", "negentien","twintig","dertig", "veertig","vijftig","zestig","z eventig","tachtig","negentig"," honderd","duizend","miljoen"]`	All unique words related to money amounts. These words are combined into one pattern that is split by the pipe symbol, which is used in the next pattern.
Money text	`(^|(?<=[^\w]))(" + Unique Text Money pattern + "+ +(en|ën| |))+( |)(euro|gulden) ($|(?=[^\w%]))`	A pattern for detecting money amounts in text format. Note that the currency is required in order for this pattern to mark these money entities.

Results

Table 19: The training and validation results for the RobBERT Regex CRF model.

	F1 Train	F1 Validation
Epoch 1	0.965	0.911
Epoch 2	0.982	0.914
Epoch 3	0.989	0.922
Epoch 4	0.986	0.926