UNIVERSITY OF TWENTE.

Justitiële Informatiedienst
Ministerie van Justitie en Veiligheid

Master Thesis

Information Extraction for Court Cases

An Exploratory Study in Information Extraction for Digitised Court Case Documents

January 6, 2021

Author:
Janko Chavannes

Supervisors:
Dr. C. Seifert (UTwente)
Dr. S. Wang (UTwente)
Dr. ir. E. de Maat (Justid)

Contents

List of Tables

List of Figures

Glossary

OCR Optical Character Recognition

NE Named Entity

NER Named Entity Recognition

NLP Natural Language Processing

CRF Conditional Random Field

1 Introduction

The Dutch **Justitiële Informatiedienst** (Justid) is a part of the ministry of Justice and Safety for the Dutch government. They manage all digital information ranging from court case documents and criminal records to fingerprints. Justid strives to deliver crucial information at the right time to help the justice system run smoothly. This is not always easy in the age of digital information, since we collect ever increasing amounts of data. People who need access to this information, such as prosecutors preparing for a court case, often have to look through many documents in order to find details about the case. During preparation they have to identify who is involved, what the case is about, where events take place, and how they are all related. Currently there is no smart system in place to help those people quickly traverse all the documents and find what they are looking for. This costs undesired time and effort.

These information extraction tasks are not new by themselves, even in the legal domain. Cardellino et al (Cardellino, Teruel, Alemany, & Villata, 2017) have done research in the legal domain, where they attempted to extract Named Entities from judgements from the European Court of Human Rights. Dozier (Dozier et al., 2010) has performed a similar task on US legal texts from different stages of a trial. They attempted to extract specific information such as judges, attorneys, courts and jurisdictions. Both of these studies look promising, however they are both focused on English texts rather than Dutch and only address part of the problem, namely the Named Entity Recognition. We are also interested in the relation between these Named Entities and an effective way of displaying this to the users.

This leads us to the following main research question:

- How can the workflow of people who need to extract information from civil court case documents be improved using Natural Language Processing (NLP) solutions?

In order to find an answer to this research question we are breaking it down into smaller subquestions. Based on the literature research, the following subquestions are devised:

1. How do modern transformer models compare to Conditional Random Field models on the task of Named Entity Recognition for the given dataset?

2. What kind of useful relationships can be found between the detected Named Entities?

3. How can the detected Named Entities and their relations be effectively represented in order to improve the workflow?

Our approach to answering the research questions and help Justid reach it's goal of delivering crucial information quickly is to build a system that incorporates different components, each addressing one of the research questions. For the development of each component the goal is not to introduce novel algorithms. Instead different existing NLP techniques are combined ranging from state-of-the-art models to traditional rule-based methods. The aim of this study is to explore how far these components can collaboratively help us in addressing our problem statement, through a series of both successful and unsuccessful experiments.

1.1 Overview

The thesis is structured as follows. Chapter 2 provides the background information and related work that is used in other chapters of this thesis.

Chapter 3 introduces an overview of the system to help the reader to get an idea of what components are included in the system and how they are related. A brief introduction to the techniques and challenges for each component is provided along with their relevance towards answering the research questions.

Chapter 4 provides details and insight into the data that was used for the development of the system. This chapter describes characteristics about the data set and what steps were taken to transform the pdf documents into data that is useful for the following components.

Chapter 5 describes the NER component of the system. The annotation process is described here, as well as the two common architectures for NER models and the experiments carried out for these models to find the most suitable one.

Chapter 6 takes you through the original plans for relationship extraction, the problems encountered during the development, and how the plan was adapted to still get useful information resembling the relationships.

These components are combined in chapter 7 where the integration of the components into one system is laid out together with the visualisation. Additionally the end-to-end experiments and results are discussed here.

Chapter 8 briefly summarises the thesis and answers the research questions that are posed in this introduction.

Finally chapter 9 mentions the practical observations and takeaways learned from working on this thesis to help future researchers in dealing with similar situations.

2 Background Information and Related Work

This chapter contains all the background information and related work that is referenced throughout the thesis. First some background information is provided for the spellchecker in the preprocessing step. Afterwards the related work for the NER step investigates the state-of-the-art models and annotation formats that will be used in this study. Finally related work is presented for the relationship extraction task.

2.1 Data Preprocessing Distance Metric

The spellchecker incorporated in the preprocessing step makes use of the Levenshtein distance, which is a metric for the similarity of two strings of text, invented by Vladimir Levenshtein back in 1965 (Levenshtein, 1966). It is commonly used to measure the similarity of words by using a few simple rules. The distance between two words is the minimum number of edits (i.e. insertions, substitutions or deletions) required to transform one word into the other.

Insertions are characters added to a string in a specific position. Deletions are characters that were removed from a string. Substitutions are characters that were replaced by another character.

$$[h]lev_{a,b}(i,j) = \begin{cases} max(i,j), & \text{if } min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + \mathbb{1}_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases} \quad (1)$$

where

$$\mathbb{1}_{(a_i \neq b_j)} = \begin{cases} 1 & \text{if } a_i \neq b_i, \\ 0 & \text{otherwise.} \end{cases}$$

a, b are the words to compare.

i, j are the lengths of the words.

The formal definition of the Levenshtein distance between strings a and b is denoted by Equation 1. Here i and j are the respective lengths of the words, and the three nested cases correspond with insertion, deletion and substitution respectively.

2.2 NER

Named Entity Recognition has been around since the 1990s. Early systems were largely rule-based and were built to extract very specific handcrafted patterns. In the early 2000s the field started to attract more attention from researchers studying machine learning models. In 2002 the first major benchmark task for Dutch and Spanish NER models was published, the CoNLL-2002 (Tjong Kim Sang, 2002), which is still often used to this day. The aim of the benchmark is to compare newly developed models by providing a common task where a model has to identify which Named Entities are present in an piece of unstructured text and categorise them as either Person, Organisation, Location or Miscellaneous. When the task was first published, the best performing model (Carreras, Màrquez, & Padró, 2002), achieved an F1-score of 77.05% on the Dutch part of CoNLL-2002. The architecture of their model was decision tree based with the (then) newly discovered AdaBoost.

Selection Criteria

For our project we selected potential NER models based on the following criteria.

- Availability of pretrained model;
- State of the art performance on the benchmark test;
- Monolingual Dutch model.

The availability of a pretrained model is important since all the best performing models on the benchmark tests are trained for a long duration using specialised hardware and use enormous general data sets (Tjong Kim Sang, 2002). Our own dataset is too small to train such a model so we would have to repeat the same training procedure on the large general data sets, for which we lack the hardware.

The benchmark performance is vital for seeing how different published models compare on similar tasks, such as the previously mentioned CoNLL-2002. Most papers include the performance of their model on these benchmarks and compare them to existing models. By examining the top performances on the benchmark we can easily find the state-of-the-art models.

Finally we need to have a model that is suitable for Dutch, the language for our data set. Many models are monolingual, meaning that they are specifically trained for a single language. However some researchers have focused their attention to multilingual models that are trained unsupervised on many different languages with the goal of making the model usable for all those languages. The best multilingual model at the time of writing is the M-BERT model from Google which performs decently well at many different NLP tasks (Devlin, Chang, Lee, & Toutanova, 2019). However Pires et al. show that the performance of this multilingual model is still considerably worse without pre-training it specifically for the target language (Pires, Schlinger, & Garrette, 2019). In the case of M-BERT, without finetuning the performance drops from 89.68% down to 77.36%. Therefore monolingual models are preferred for the time being.

Selected Models

Over time, many improved models have been published using more sophisticated architectures. A Conditional Random Field (CRF) model developed at Stanford University (Finkel, Grenager, & Manning, 2005) was trained for English NER and performed quite well on that task. Moreover, the same researchers found that training their model with identical features used by the English model, worked decently well for other languages. The English version had an F1-score of 86.31% on the English CoNLL-2003, while the Dutch version on their model, which uses the same features, reached an F1-score of 79.71% on the CoNLL-2002 set [1]. Both of these models from Stanford University are often used as a baseline to beat for newly released papers. This makes the Dutch model a good choice for one of the models to evaluate for our research. Earlier this year Stanford has releases a new project called Stanza (Qi, Zhang, Zhang, Bolton, & Manning, 2020). This project contains an updated version of the Dutch Stanford NER model, boasting a significantly improved

[1] `https://nlp.stanford.edu/projects/project-ner.shtml`, last accessed 2020-10-12

F1-score of 89.2% on the CoNLL-2002 task [2]. This will be the first NER model used in this thesis.

Most recent publications in the field of NER incorporate a new state of the art architecture, the Transformer model architecture. The results of these models look very promising for various NLP tasks and for different languages. The best Dutch transformer model as of writing is BERTje (de Vries et al., 2019). This model was also evaluated on the CoNLL-2002 task and achieved an F1-score of 90.24%, which is slightly better than the previously mentioned model for this task. BERTje is the second model that will be evaluated for the NER component of our system.

Transformer Models

The Transformer model architecture, which was introduced in 2017, has gained a lot of attention from researched in the field of language modelling. It is based on techniques used in more traditional Recurrent Neural Network models, however it has a simpler internal structure and more effective way of dealing with token positions (Vaswani et al., 2017). This allowed transformer models to achieve similar or even higher performance on certain tasks such as machine translation, while being an orders of magnitude faster in training.

Soon after the publication of the transformer model, it was incorporated in a new architecture called BERT, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (Devlin et al., 2019). This model further improved the conceptual language understanding of the model by incorporating the context on the right side in addition to just the left side. According to the researchers the bidirectional nature of the model combined with the positional encodings of the transformer allows it to gain a deep understanding of the language, hence why it performs well on so many different tasks while using the same model structure. In the paper where the original BERT model was published, it already posted state-of-the-art results on eleven NLP tasks, however NER was not yet one of them. Chapter 5 shows a BERT based model on our dataset and how this language model might affect the results.

[2] `https://stanfordnlp.github.io/stanza/performance.html#system-performance-on-ner-corpora`, last accessed 2020-10-12

Tagging Formats

In order for the models to be evaluated on our dataset, we also need to manually tag the NEs that are present in the documents. There are many different annotation tools available with different advantages and disadvantages. Our dataset contains classified information, so any web-based tool is out of the question. There are also paid tools, which can be expensive, for this research we stick with a free open-source option. The final option that was used in this research is the Brat rapid annotation tool (Stenetorp et al., 2012). Brat is little older than the alternatives and has a dated UI, however it's simple to work with, free, can be run locally, and does not have any proprietary export formats. The annotations format for the tool is a simple text span with a start index and end index for each annotation, which can include multiple words.

Table 1: Example of the IOB annotation format for single and multi word NEs.

Token	Label
Mark	I-Person
works	O
in	O
The	B-Location
Hague	I-Location

Aside from the text span format, another common annotation format is the *IOB* format introduced in 1995 (Ramshaw & Marcus, 1999). This format labels every token in the text with either **O** if they are not part of a NE, or prefixes their original category label with **B-** or **I-** if they are part of a NE. Contrary to the span format, the *IOB* format does not indicate a multi word NE directly. A multi word NE can then be encoded by prefixing it's first word with **B-** (beginning), and all subsequent words with **I-** (inside). Single word NEs are always prefixed with **I-**. Table 1 shows an example of this for both a single word and multi word NE.

2.3 Relationship Extraction

Relationship extraction is a research field that is still evolving rapidly. There are many different approaches from neural models to pattern based approaches. Often models are highly specific to the task for which they were

developed. These tasks can have varying degrees of granularity and be for different domains. TACRED is one of the largest such tasks with a corpus containing over 100K news articles. For this tasks models have to identify 42 fine-grained relations such as *place of birth* or *religious affiliation*. Zhang et al (Zhang, Zhong, Chen, Angeli, & Manning, 2017) showed carefully designed neural models can get up to 65% F1 score, where the recall and precision are almost equal. Pattern based approaches score higher on precision with over 80%, however they have very low recall of 23% leading to a lower overall F1 score of 35%. Traditional simple models combined with such patterns can achieve an F1 score which is still lower but much closer to the neural models. Additionally they have higher precision than the neural model. Overall to get the best performance, a neural model is the best approach.

SemEval is another popular repeating relationship extraction task for a smaller corpus of around 10K examples which focuses on detecting 9 more general semantic relations such as *Content-Container* and *Cause-Effect* (Hendrickx et al., 2010). Here we can see that neural models again perform the best when measuring F1 scores. Since this tasks involves fewer output categories, we can expect better performance from the models and indeed the best F1 score for this tasks is considerably higher at over 80%.

3 System Overview

This chapter describes the overview of the system as a whole and gives a brief introduction to each of the components in the system. First an artificial example is given that will illustrate how the system works and what the relevance of each component is towards answering the main research question. Afterwards, different techniques and challenges are laid out for each component on a high level. More detailed information about the separate components is provided in later chapters.

3.1 System Walkthrough

This section takes you through the components of the system, shown in Figure 1, on an abstract level. An artificial example will be used to go from the input documents to the final visual representations as the output. At each stage, the function and relevance of the component is briefly discussed.

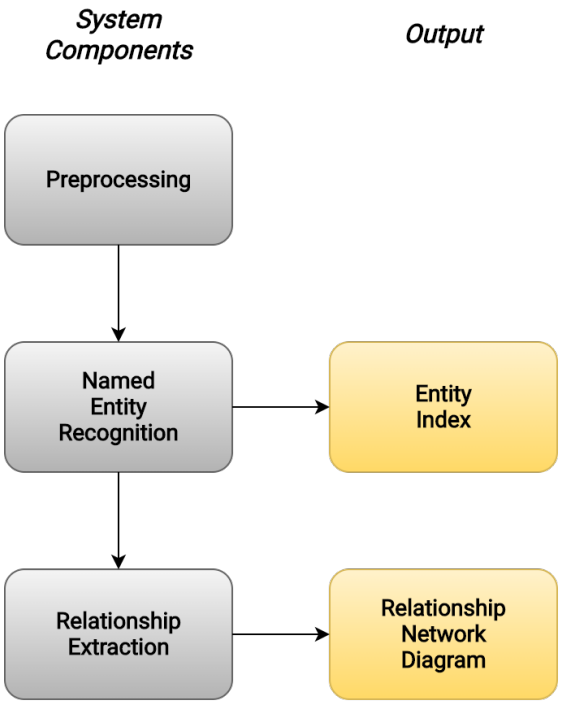

Figure 1: High level overview of the different system components.

A court case, consisting of one or more documents, is loaded into the system and first passes the *Preprocessing* component. This step attempts to make the input more suitable for the following components. It will correct errors

that were introduced when documents were scanned and transformed into text, or from encoding and decoding special characters.

The preprocessed data is then fed into the *Named Entity Recognition* component. This step will extract important Named Entities such as person, locations and organisations from the documents. In doing so, all information regarding who is involved in the case and where things take place can be extracted. After the NER step is done, enough information is available to start building an *Entity Index* as the first output. This index provides a reference between mentions of the Named Entities and in which documents they occur, which is helpful when someone wants to find information about particular persons for example.

Now that the Named Entities are known, we know who is involved in the case, however we don't know how yet. To get a deeper understanding of how the found Named Entities are related, *Relationship Extraction* is used. This technique serves to identify certain relations between two Named Entities. Depending on the model these can be specific relations such as "organisation A employs person B", or more abstract relations such as "object X is contained in object Y". For our goals we want to identify relations relevant to court cases, such as "person A is a lawyer of person B", or "person C is family of person D". After extracting these relations, the final output in the form of a *Relationship Network Diagram* can be generated. This diagram will visually show how the found entities are related to each other in order to provide a quick and concise overview of the case without reading any of the documents.

3.2 Techniques and Challenges

This section describes for each of the components in the system what kind of techniques and challenges there are.

Preprocessing

There are many different preprocessing techniques, ranging from organising texts to simplifying it. Organising tecniques could be splitting the paragraphs of a text into individual sentences or tokenizing all the words. Simplification techniques strive to reduce the complexity of a text while simultaneously keeping the important information. This can be done by converting occurrences of a verb to its base form (e.g. *walking* to *walk*) or removing com-

mon stopwords (such as *the* or *in*) amongst other methods.

The main challenge for preprocessing is to choose the right methods that allow a model to perform better. We want to make the input as simple as possible for the model while keeping the important information. For example, a common form of preprocessing is converting the entire text to lowercase, however for the NER steps this is not suitable since casing is important for identifying names of persons, organisations, and locations.

NER

For Named Entity Recognition there are two primary approaches, a manual approach or a machine learning approach. In general a manual approach can have better precision since it is specifically crafted for the dataset, however it takes a lot more effort to create all the patterns and requires more domain expertise. On the other hand recall is often lower for this approach since the patterns are not exhaustive as we have seen in chapter 2.2. The other main approach is to use machine learning in order to detect patterns. With this approach, all the Named Entities in the text have to be manually annotated an the model can learn to detect patterns in the dataset. This requires less domain knowledge and time, however it does require more data. Additionally these types of models generalise better to future data, opposed to handcrafted patterns, which leads to higher F1 scores. Since our main metric is F1 score, our time is limited and I am no expert in the legal domain, the machine learning approach is more suitable.

The challenge for this component is to find out which type of model is most suited for the dataset and also provides the right output. Training a new model requires a lot of data, more than we have here. However, there is enough to evaluate the performance of existing models or potentially apply transfer learning to make an existing model more specific to our dataset. As for the output, there are many different open source models available which are trained for different tasks, such as identifying different proteins in medical texts, or detecting general Named Entities such as persons. When using an existing model it is important to make sure the output is relevant to our goals.

Relationship Extraction

Similar to NER, relationship extraction can also be done either manually or by machine learning. The same advantages and disadvantages for the approaches apply here, hence the machine learning approach is more suitable for this component too.

The challenge for this component is again to find a suitable model, however there is even more variance in the output. Whereas NER can detect general entities, the relationships can vary a lot more. For example when the NER step detects a person and organisation in one sentence, a relation between them can be that the person founded the organisation, the person works for the organisation, the person is the head of the organisation, the person is a customer of the organisation, and many more.

Visual Representation

All of the information that is extracted is not useful until it can be understood by the user of the system. There are different ways to provide information in a diagram, for example by adding text and colours, changing the layout of the graph or grouping certain nodes together. All of these methods can convey more information and add to the overall graph importance.

The main challenge with this visual representation is to strike a good balance between providing the important information to the user, without overloading them with too much information. If there are too many nodes, colours and text it is no longer possible to quickly see what is going on in the diagram.

4 Data Set

This chapter investigates the data set that was used for building the system. First the origin and the type of data is laid out with some initial properties such as the size. After that some preprocessing steps are examined to turn the original documents into useful input for developing the other components of the system.

4.1 Data Set Description

The data used in this project consists of court cases from the Dutch justice system regarding civil law family cases. The subject of the cases ranges from self-harm and mental disorders to domestic violence. These court cases contain a number of scanned pdf documents divided into 13 categories shown in Table 2, the id's from this table are used throughout this chapter to indicate the document categories.

Table 2: Document categories Dutch and English

id	Dutch term	English term
1	Correspondentie over procedure	Procedural correspondence
2	Deskundig rapport	Expert report
3	Interne documenten	Internal documents
4	Intrekking	Withdrawal
5	Oproeping	Subpoena
6	Pleitnota	Appeal
7	Proces verbaal van de zitting	Report of the hearing
8	Processtuk	Process piece
9	Rechterlijke uitspraak	Court ruling
10	Toevoeging	Supplement
11	Verweerschrift	Defense
12	Verzoekschrift	Petition
13	Zittingsaantekeningen	Hearing notes

The dataset contains 59 court cases with each case containing a multitude of documents, for a total of 619 documents. The documents are relatively clean scans, however most of the documents contain at least a bit of handwritten text, ranging from a signature or stamp, to annotations and attachments which have also been scanned. Some of the documents are exclusively

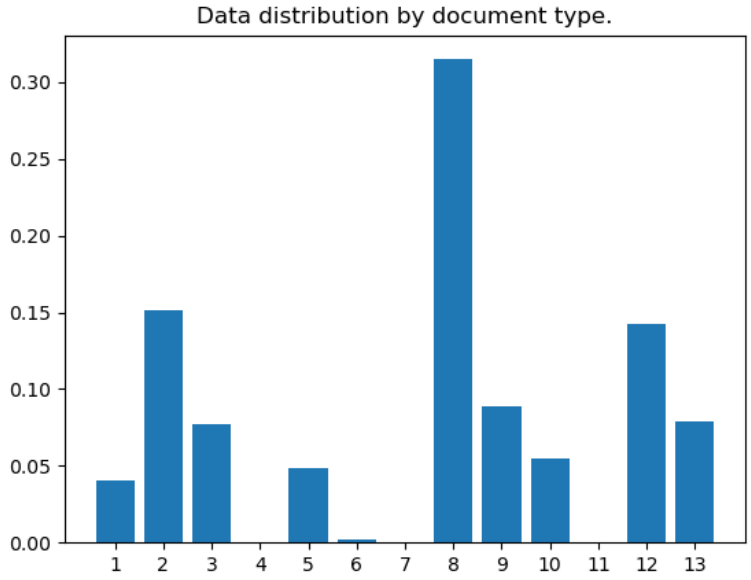

Figure 2: Data distribution per document type

handwritten. Optical Character Recognition (OCR) has already been applied to the data in order to transform the scanned image back to text, however the performance for handwritten content is poor. An example of this will be shown later in this chapter.

The types of documents in this dataset are not represented equally. Certain types will occur many times in each case, while others may occur only in specific cases. Figure 2 shows the document type distribution among the entire dataset. The layout of documents in each type can also vary wildly. For example the type *interne documenten* (internal documents) often contain (handwritten) memo's, e-mails and letters.

4.2 Data Preparation

The data consists of pdf files from the scanned documents. First the raw text had to be extracted from the OCR layer of these pdf files. This was done using Python 3.6 [3] and the pdfminer.six [4] package.

After extracting the text from the documents, it was not yet ready to be

[3] `https://www.python.org/`, last accessed 2020-10-09
[4] `https://github.com/pdfminer/pdfminer.six`, last accessed 2020-10-09

used as is however. As mentioned before, the OCR system didn't handle handwritten text well to the point that none of the original handwriting could be retrieved. To illustrate the problems that arose from this, see the following sample text which is an excerpt of one of the documents containing a form that was filled in with handwritten text. The printed parts of the form are largely recognised correctly, however none of the written answers could be retrieved. For the record, the sample picked was not random, it is the most neat handwriting we could find in all documents. These texts are not recognisable for a human let alone a model, so we attempted to improve the data by correcting errors using a spellchecker.

```
Gegevens advocaat
Voorkeursadvocaat:        (7;4  Nee            E la
Naam •  Ç .          V1/4:-.1ƒ-1  \--k o..a.

Registratienummtr :
...  r‘
,’-..,(2.,:n.-,         .J.:j\i.OL(,:i27,Q.,

..r.           f

Kantoornaam •
Postcode / Plaatsnaam •  \-1,  UI r)k,      —\----i’
Telefoonnummer •  0.5/C  - 12.5555
```

Error Correction

In order to correct the errors resulting from OCR, we applied a form of spellchecking that attempts to correct words that are unknown to the spellchecker. The vocabulary of the spellchecker, which contains over nine million words, was built using collection of Dutch Wikipedia pages [5]. The documents were then put into the spellchecker which processes the text word by word and calculates the Levenshtein distance (explained in chapter 2) between that word and words in the vocabulary. Words that occur in the vocabulary are considered correct and remain unmodified. Words that do not occur in the vocabulary are matched against the words that are. The spellchecker

[5] https://dumps.wikimedia.org/nlwiki/latest/, last accessed 2020-04-30

attempts to find the most common word from the vocabulary that has a Levenshtein distance of one to the checked word. The original word is then corrected by the found word. If no words are found, the process is repeated for a distance of two. When there is still no compatible word found, the original word is replaced by an unknown token placeholder. The results of the spellchecking per category can be found in Figure 3.

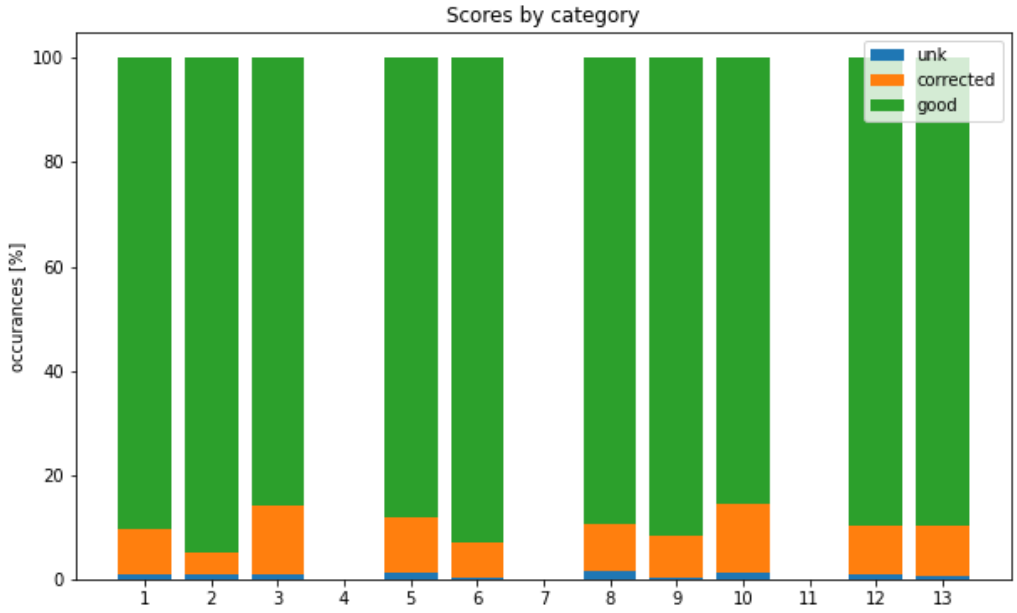

Figure 3: Relative spellchecking results per category, categories with no documents have empty bars.

Due to the way the spellchecker is set up, corrected words are more likely to to be the same as the original word from the pdf document, however this is not guaranteed. The spellchecker may also introduce some new errors, such as substituting a wrong word for another wrong word, or changing a correct word to a wrong word. Despite these new errors, the resulting texts are closer to the original text based on manual inspection. It is hard to quantify this though, after all if we had the original text to compare the result to, there would be no need for a spellchecker.

Overall the documents with the least corrections are closest to the original counterpart. For that reason the category that required least corrections (i.e. *Expert reports*) is chosen as the basis for developing the rest of the system. Additionally this category also contains the most data out of all categories as can be seen in Figure 4, and more data means that training and evaluating

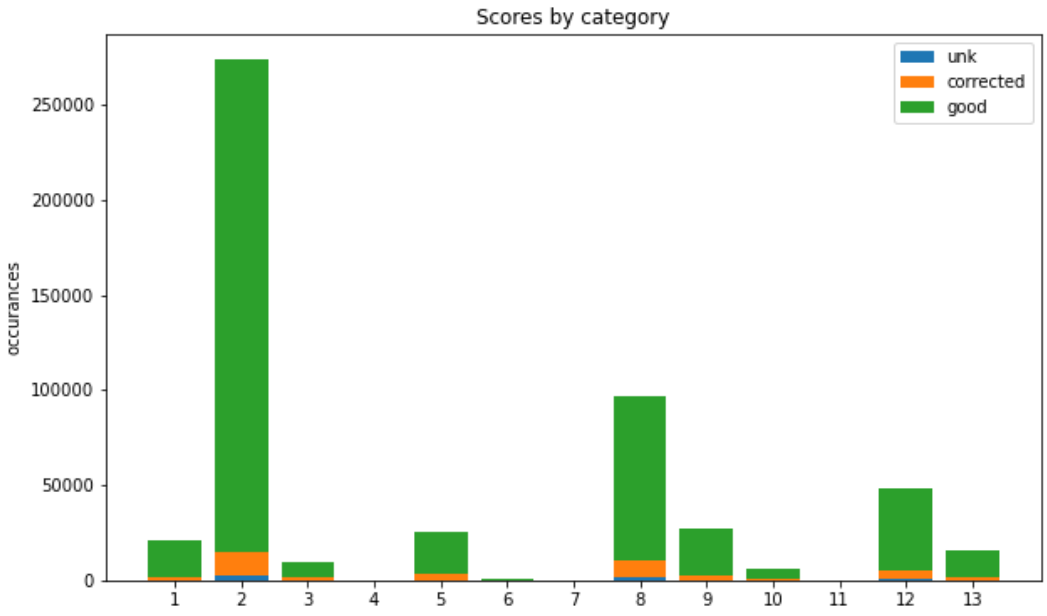

Figure 4: Total spellchecking results per category, categories with no documents have empty bars.

models will be easier. The other categories (i.e. **not** *Expert reports*) are not used in later parts of the report.

5 Named Entity Recognition

This chapter investigates the Named Entity Recognition (NER) component of the system in order to answer research question 1. First, section 5.1 shows the experiments performed to determine the which of the models introduced in chapter 2 performs the best on the dataset. Additionally this section describes how the data was prepared and annotated. In section 5.2 the results of the experiments are shown. These results are then discussed in section 5.3 to determine which model will be used in the combined system. This section also discusses some of the interesting findings an the attempts made to address some of the problems encountered during the experiments. Finally the conclusion in section 5.4 answers the research question.

5.1 Experiments

This section describes the process of evaluating the existing potentially suitable NER models mentioned in section 2.2, namely the CRF model *StanfordNER* and the transformer model *BERTje*. First the annotation method is briefly explained followed by the scoring method for each of the chosen models. Finally some of the most common mistakes for both models are highlighted.

Annotating The Data

In order to evaluate the performance of the NER models for our data set, the NEs first needed to be annotated. Recall from section 2.2 that the existing models predict one of four different labels; Person, Location, Organisation and Miscellaneous. For our research we are only interested in the first three, so Miscellaneous NEs were not annotated. Words in the text that were not annotated as Person, Location, or Organisation, were implicitly marked as category Other. In short the NEs belonging to one of the following categories were annotated using the Brat rapid annotation tool [6]:

- Person;
- Location;
- Organisation;
- Other.

[6] `http://brat.nlplab.org`, last accessed 2020-09-10

Table 3: Annotation counts per NE category

Category	Count
Person	2646
Location	823
Organisation	578

The annotations were all done by myself. While I am a native Dutch speaker and the texts are in Dutch, I am no expert in the field of medical and legal texts. Therefore the annotations may contain errors, especially regarding the field specific organisations and abbreviations. The total number of annotations per category can be found in Table 3.

Scoring the Performance

The metric of choice for the evaluation of the models is the ubiquitous F1-score which is used in nearly all papers about NER models. Since the F1-score is a compound metric based on recall and precision, these two were also computed. Additionally recall and precision give a deeper insight into the predictions and how balanced these are for the different categories. The recall shows what fraction of the annotated NEs per category were retrieved and predicted as the same category by the model. The precision indicates what fraction of the predicted NEs were also marked as the same category in the ground truth annotations.

Aside from the metric there are also different ways of judging whether the output of the model is correct compared to the ground truth annotations. For this research we have chosen to look at both partial matching and full matching. With the full matching approach for both single and multi word NEs, any NE that the model outputs is considered correct if the entire NE lines up with the ground truth annotation and is of the same type.

The partial matching approach is the same for single word NEs. For multi word NEs the output of the model is matched against the ground truth and is considered correct if they are both of the same type, and any of the individual words overlap. This type of matching is useful to provide an upper bound on the model performance. Additionally, sometimes only a part of the NE has to be recognised for it to be a useful prediction. For example, if the ground-truth is *Politiebureau Zwolle West* and the model detects *Politiebureau*, that could be

the essential information that we are looking for.

5.2 Results

This section presents the results and observations for the NER experiments. First some of the common recurring errors observed for both of the models are listed. This is followed by the general performance of the models according to the evaluation metrics.

Common Mistakes

Both of the tested models made mistakes with either of the matching approaches. Here we will identify some of the common recurring mistakes for either of the models and divide them into three types of errors; false positives, misses and phantoms.

False positive errors are predictions of the model that have the correct NE except with the wrong category. For example a location such as *Amsterdam* predicted as a person.

Misses are NEs which are marked as a ground truth annotation, yet they were not recognised by the model. In this case important information belonging to one of the three NE categories is missed entirely by the model.

Phantom predictions are predictions by the model for pieces of text were not marked as a NE in the ground truth annotations (i.e. category Other). Here the model mistakenly marks irrelevant information as being important information.

The common mistakes for BERTje are listed in Table 4, while Table 5 shows the common mistakes for StanfordNER. These tables show per prediction category and type of mistake what the most common errors were. Finally the main differences for partial matching opposed to full matching are shown.

Table 4: Common NER mistakes for BERTje

	Person	Location	Organisation
False Positives			
	Loc (e.g. *Zwolle*)	-	-
Misses			
	Formal names (*Surname, Initials*)	Regular locations (*Zwolle*, *Hardenberg*)	Specific organisations (*Tactus*, *Dimence*, *Trajectum*)
	Double surnames (*Name1-Name2*)		
Phantoms			
	Postal code letters	Loc in org name (*GGZ **Zwolle***)	Unknown word token (<*UNK*>)
	Other random 1,2,3 letter words	(Part of) Multi-word streetnames	*U* (Eng. formal *You*)
Partial matching			
	Parts of multi words NEs in general. (e.g. *'Dimence West Overijssel'* to *West* or *Overijssel*		

Table 5: Common NER mistakes for StanfordNER

	Person	Location	Organisation
False Positives			
	Person names in locations	Care organisations (*Dimence*, *Trajectum*)	Addresses (i.e. *Street 1*)
Misses			
	Formal names (*Surname, Initials*)	Regular locations (*Zwolle*)	Specific organisations (*Tactus*, *Dimence*, *Trajectum*)
			Orgs containing extra words (e.g. *Politiebureau Kogelman*)
Phantoms			
	Abbreviations (e.g. *Dhr* or *Dhr.*)	Partial loc names	Rooms and departments inside organisation
		Postal code letters prepended to location	Unknown word token (<*UNK*>)
Partial matching			
	More formally written names	-	-

Table 6: NER results partial and full matching per category.

	Precision				Recall			
	Per	Loc	Org	Other	Per	Loc	Org	Other
Partial matching								
BERTje	41.4%	41.9%	8.8%	99.8%	81.9%	64.4%	39.3%	98.5%
StanfordNER	**56.4%**	**67.2%**	**15.5%**	**99.9%**	**85.4%**	**77.5%**	**43.4%**	**99.2%**
Full matching								
BERTje	5.7%	21.8%	1.9%	99.3%	15.7%	31.9%	1.0%	98%
StanfordNER	34.4%	58.8%	13.8%	99.6%	57.5%	69.2%	37.8%	98.9%

Table 7: NER results summary (micro-averaged), partial and full

	Recall	Precision	F1
Partial matching			
BERTje	70.5%	37.1%	48.6%
StanfordNER	**77.2%**	**55.1%**	**64.3%**
Full matching			
BERTje	19.9%	11.1%	14.3%
StanfordNER	59.5%	41.0%	48.5%

Model Performance

In Table 6 we can see the detailed precision and recall per class for both models. Table 7 shows the micro averaged results. From these tables it is clear that the StanfordNER outperforms the BERTje model for both types of matching, more on that in the discussion. Additionally, as expected there is a significant drop in performance for both models when scoring with full matching. The effect of this is more severe for BERTje. The reason for this is that this model often only gets a part of the answer correct, as can be seen from the common mistakes. StanfordNER also makes more mistakes when applying full matching, however the effect of this is not as profound.

5.3 Discussion

As seen in section 5.2 there are some common recurring mistakes for both models. This section addresses some attempts made in to reduce the number

of mistakes along with potential solutions for future development.

The organisations in the dataset are domain-specific and likely were not present in the original training sets for the models. One thing to note is that while these organisations are specific to the data, many of the documents refer to the same organisations. This allows for a white list of organisation names which can be used instead of, or in addition to, the existing model.

Another thing that can be observed in the common mistakes was that the unknown word token <*UNK*>, an artefact of the spellchecker, would sometimes be included in the predictions as a NE. These misclassifications have been corrected for the results by ignoring all of the unknown word tokens from the models outputs. They were still in the input for alignment purposes between the ground truth and the output.

Abbreviations were also a problem for both models. They were often not recognised as abbreviations for common words, that do not actually represent a NE. We can see this with the commonly mistaken *Dhr* short for *De heer* (respectively *Mr* or *Mister* in English). A large portion of these results can be filtered out by finding creating a blacklist based on the common errors and removing all model predictions that are on this list. A small blacklist containing the most common abbreviations is already implemented for commonly used Dutch words such as the example. This list can be extended in the future by people who have more knowledge of the medical and legal domains.

The final recurring mistake is with location predictions, where sometimes the final letters of a postal code (format *1234 AB* in The Netherlands) are included with the city name. This causes the prediction to be slightly incorrect, and more importantly makes it more likely for the model to misclassify it as an organisation. Additionally since it was not recognised as a location, this produces some additional miss errors. A solution is implemented in post processing to correct for overlapping postal codes. Two ways of correcting for postal code overlap have been examined. The first method removes all detected NEs that overlap with postal codes. The second method was to strip all parts that overlap with postal codes from the prediction, this resulted in the best performance and is used in the final model. Table 15 in the appendix has more detailed results for the effectiveness of the three methods.

One final thing to note is that while BERTje performs slightly better than StanfordNER on benchmark tests, the performance of BERTje for our data is considerably worse especially for full matching. The most common errors were in the form of small words of up to three letters, which resulted from a combination of the spellchecker, OCR, as well as actual short words such as abbreviations in the text. Initially this seemed to be the biggest problem for BERTje and we evaluated the performance of the model on documents that were cleaned by hand. This cleaning process removed junk characters, replaced unknown words with the original words, and matched the casing for all words. As a result the cleaned documents contained the exact text from the pdfs. This resulted in approximately 5 percent point increase for BERTje using the full matching approach, which does not fully explain the low performance. More detailed statistics can be found in Table 14 in the appendix. Even with this performance increase, StanfordNER is still better than BERTje for our data set.

The cleaning step covered errors, however the sentence structure for parts of our documents are also different to what the model is trained with. For example at the start of each document or in forms, there are many lines with facts such as "Name: xxxxx" or "Documentnr: xxxxx" which are not really sentences. In chapter 2.2 we saw that the BERT models require very large datasets to train. Additionally these consist of high quality full sentences from sources such as books and newspaper articles. The authors of BERTje even specifically mention that they removed sentences originating from chats or Twitter for being too low quality (de Vries et al., 2019). Similar to training, the BERT based models are also evaluated on the same type of high quality data. Our final hypothesis for the performance gap between the benchmark and our dataset is therefore that a combination of domain specific abbreviations and jargon, combined with the sentence structure might disrupt the internal language modelling of BERTje.

5.4 Conclusion

In this chapter we saw two state-of-the-art models from two different architectures that were introduced in chapter 2.2. In order to answer research question 1 both of the models were evaluated, and based on the most frequent errors different pre- and post processing methods were examined in order to improve the performance. Both models were unable to match their bench-

mark performance on our dataset for any of the categories. Organisations seemed particularly hard for the models due to a combination of domain specific jargon and many abbreviations which were mistaken for organisations. While on the benchmark task both models had approximately the same performance, in our experiments BERTje scored considerably worse even after all the pre- and post processing methods were applied.

The research question that this chapter set out to answer is *How do modern transformer models compare to Conditional Random Field models on the task of Named Entity Recognition for the given dataset?* In our experiments the best CRF model performed better compared to the best transformer model. BERTje is only one instance of a transformer model, and even though it scored the best on benchmark tasks we can't definitively say transformer models are worse for the tasks. However, BERTje employs the same type of training and the same model architecture as the original BERT model and its variations. It is therefore not unlikely that other general BERT variations would also suffer from similar performance loss.

To summarize, with respect to our research question we can conclude that for the instances of the models we tested, the StanfordNER CRF model compared favourably to the BERTje transformer model and it is likely that StanfordNER will outperform other general BERT based models.

6 Relationship Extraction

This chapter investigates the relationship extraction component of the system. First, the process of annotating, developing and evaluating the relationship extraction component of the system with our initial approach is laid out in section 6.1. This section shows the relationships that we are interested in finding, the steps taken to annotate and prepare the data, and some issues encountered during implementation. Unfortunately due to these issues this approach turned out to be unfeasible. Instead that approach was adapted to further classify Person NEs into more detailed subcategories, indicating how a Person NE relates to the subject of the case. This way, we could still extract important information resembling the original goal. Section 6.2 describes the adaptations that were made and what the final approach was for the implementation of the relationship extraction component. Section 6.3 shows the results and observations about the experiments. Section 6.4 discusses the problems that were encountered with the final approach and some potential improvements that can be made to this part of the system.

6.1 Initial Approach

Initially the plan was to attempt to extract important relations between NEs using existing generalised relationship extraction models. Based on what we encountered when examining the documents, and the wishes of Justid, we composed a list with types of relations that could be useful to provide insight into a case. These types can be found in Table 8.

The relation *Guardian* is a relation between a guardian and a child. The guardian is someone who raises and houses the child and is usually a family member, though never the parent.

Table 8: Relationship types for initial approach

Relationship	Type source	Type target
Parent	Person	Person
Spouse	Person	Person
Guardian	Person	Person
Caretaker	Person	Person
Works for	Person	Organisation
Based in	Person	Location/Organisation

Table 9: Annotation counts per relationship

Relationship	Count
Parent	1
Spouse	4
Guardian	22
Caretaker	36
Works for	30
Based in	57

The relation *Caretaker* is broader in our context, it can describe a nurse taking care of a patient in the usual sense of the word. Other relations that are included under this type are a mentor or teacher who offers a supporting role to a child.

Based in is used mostly used for people and their homes. Other uses for this relation type are to indicate care organisation or locations where people are kept temporarily. The definitions of *Parent*, *Spouse*, and *Works for* are straightforward.

The annotation process is done similar to the annotation process for NER, described in section 5.1. We used the same dataset and existing NE annotations to mark our defined relations. The annotation results for these relations can be found in Table 9. Unfortunately, while we had thousands of annotations for the NER component, we had much fewer annotations for the relationships. The largest category only contained 57 examples and some categories not even a handful.

The low number of relations compared to the NER annotations is not entirely explained by a lack of relationships in the documents directly. One of the reasons is that many models require short pieces of text, usually at the sentence level, and in our data the NEs involved in a relation were often further apart. For example, information about the client is given at the start of the document and treatment by a nurse is provided halfway. In this case the nurse is a caretaker of the client, however we cannot annotate this as such.

Another reason which is related to the first, is that we can only annotate relations between two annotated NEs. However, many of the relations that were in the text were between one actual NE and a **reference** to another NE

(such as "her", "the patient", "the doctor" etc.). Now there is a research field within NLP, called Coreference Resolution, which deals with resolving these references (such as "him/her") back to their actual NE. We felt that at this stage it would take too much time to incorporate such methods, so we opted for another approach instead.

6.2 Final Approach

In the initial approach we listed the relations that were of interest. Many of these relate back to the subject of the court cases. Additionally when annotating, we observed that the context surrounding the mentions of Person NEs often contained information about their relationship to the subject. For example, doctors or psychiatrists often had their job title in the same sentence as their name. With these observations, combined with the fact that we already had 2646 Person annotations from the NER component, we decided to change the approach to further classifying the role of each detected person NE based on the context surrounding their annotation. Of course everything in the document is related to the subject in some way, however from the context it is clear that this relation is mostly direct (e.g. if it is mentioned that someone is a caretaker, they will be caring for the subject). So we make use of the following assumption:

- The roles of people mentioned in the document reflect direct relations to the subject.

We wanted to keep a fine level of granulation for this approach just like in the initial approach. After redoing the annotations for all the Person NEs, we found that some of the categories still lacked examples, as can be seen in Table 10.

The low number of examples for some of these categories lead to the same issues discussed before. Finally some of the categories were merged together, in an attempt to make the categories more balanced while simultaneously preventing categories from being too general. The new categories along with their annotation counts and which old categories they contain can be found in Table 11.

About the Model

The new categories still do not provide a very large training set, however it is enough to train simple models. For this reason we opted to train a Naive Bayesian classifier since it does well with small data sets and is often used for text classification tasks (Jurafsky & Martin, 2009).

Recall that the context often contains information about a Person NE, however it needs to be transformed into a form that can be used as an input to the model. The transformation was done with a *Word Vectorizer* which takes N words in front of the Person NE and encodes them into an array of word counts for the most common words. Similarly the N words behind the Person NE were encoded into a second array. These arrays were then concatenated into one longer array which serves as the input for one sample.

A Naive Bayesian model itself does not have hyperparameters to tune, however the word vectorizer can be optimized. Multiple values were considered for the maximum number of features (i.e. the length of the arrays) that the vectorizer considers as well as the number of words from the context N. A higher maximum number of features can improve performance by including words that are less common and more specific to certain categories. However, performance can also decrease since words that are less common can be random and not actually contain information specific to a certain category. On a similar note, a lower maximum number of features might decrease the performance by excluding uncommon words specific to a category, or it can increase

Table 10: Annotation counts for person fine grained subcategories.

Category	Count
Grandparent	5
Parent	69
Sibling	90
Child	3
Spouse	4
Client	2032
Doctor	193
Nurse	201
Relative	13
Other	37

Table 11: Merged person subcategories with annotation counts.

Initial category	Final category	Count
Grandparent Parent Sibling Child Spouse	Family	171
Client	Client	2032
Doctor	Doctor	193
Nurse	Nurse	201
Relative Other	Other	50

the performance since it excludes the random words that carry no important information. Hence this is one of the parameters that will be optimized.

The length of the context N also needs to be optimized. A small N includes words that are very close to the NE which makes it more likely to include relevant words only and improve the performance of the model. On the other hand, sometimes the important words are further away in the sentence and a small range will exclude these. A larger N will capture the important words further away, however it might also capture important words that actually belong to another Person NE. Therefore we have to determine what value for N is the best.

The preprocessing, training and evaluation was done using *Python 3.6.1* and the *scikit-learn* [7] package. The total data set was split into a train and test set containing 85% and 15% of the samples respectively.

6.3 Results

This section shows the results from the experiments that were conducted in the final approach. Table 12 shows the performance of model for different configurations of the max frequency and context length N. We can see from the table that a maximum frequency of 500, combined with N of 5 yields the best result.

[7] `https://scikit-learn.org/stable/`, last accessed 2020-9-29

Table 12: Performance for different model configurations micro averaged.

Max features	N	Precision	Recall	F1
250	5	0.81	0.83	0.82
500	5	**0.84**	**0.84**	**0.83**
250	10	0.81	0.81	0.81
500	10	0.79	0.79	0.79

An interesting point to note here is that for a lower maximum frequency, increasing N leads to a better performance. On the other hand, when the maximum frequency is 500, the performance decreases for larger N. This is likely due to a combination of consequences for both of the variables that were mentioned in the previous section. The words further away from the NE are more generic words (i.e. words that hold no information about the Person NE), simultaneously more of those words are being captured due to higher maximum frequency. It appears that for smaller N, capturing words closer to the NE, a higher maximum frequency leads to more important words being included.

More detailed results, including the performance per category, is laid out in Table 13. Here we can see the results vary considerably per category.

The *Client* category achieved the best results overall, which is to be expected since it has the most examples to learn from. Additionally, since it is larger than the other categories, the model has a slight bias towards *Client* because it occurs more often.

The category *Doctor* stands out as it performs better than other classes with roughly the same number of samples. This is likely because it is common for academic titles or job titles to be close to the NE in this category. For example, a pattern such as "[...] Dr. Appelman, (**Psychiatrist**) [...]" occurs often.

The smallest categories, *Other* and *Guardian*, achieved relatively low scores, which is expected since they have fewer examples.

The *Family* category also scores very low. Recall from the previous section that this category is composed from 5 different categories. It is likely that

Table 13: Person classification results for the best model evaluation.

Category	Precision	Recall	F1	Samples
Family	0.70	0.44	0.54	32
Doctor	0.86	0.70	0.78	27
Nurse	0.61	0.69	0.65	29
Client	0.88	0.94	0.91	302
Guardian	0.64	0.54	0.58	13
Other	1.00	0.12	0.22	8
Overall [a]	0.84	0.84	0.83	411

[a]Micro-averaged

there is more variance in this category because of this, and the model requires more samples to learn all the patterns in this category.

Similarly the category *Other* suffers from the same problem, since it contains all samples that do not belong to any of the other categories which can lead to a lot of variance.

6.4 Discussion

The results obtained for the relationship extraction component are not bad overall, although we can clearly see some variance between the categories. As mentioned before, this be attributed to an imbalance between the different categories as well as a relatively small set of examples. This problem has been partly addressed by combining some of the smaller categories together into a bigger category. As a result the distribution over the new categories as well as the minimum number of samples improved.

Another technique that can be used for handling the class imbalance is undersampling the biggest category, or oversampling the smaller categories in order to get approximately the same number of samples for each category. Both techniques can achieve the same goal, however given that the total dataset is not very large, it is likely that oversampling will work better as this increases the total number of samples.

6.5 Conclusion

In this chapter we explored relationship extraction methods in order to answer research question 2. The initial plan was to use generalised relationship extraction models to detect relations between the different Named Entities. Unfortunately after annotating many of the relationships had very few occurrences, too few to meaningfully evaluate existing models let alone to train a new model on. Instead we altered our method slightly and instead focused on determining the role of each person NE in relation to the subject of the documents.

The new approach yielded significantly more samples so a new model could be trained to predict the person roles. For some categories that occur more frequently this works quite well, for categories that do not occur that often or are very broad the performance is not so great. In short, to answer the research question: *What kind of useful relationships can be found between the detected Named Entities?*, the system can detect whether a person is the client or family, a doctor, a nurse, or a guardian of the client.

7 Integrating all the components

This chapter discusses the integration of the components into one system. First the architecture is laid out where some of the design decisions are explained. Afterwards the details about the visualisation diagrams are mentioned. Finally, a case study for the output of the system along with the results and discussion is presented.

7.1 System Architecture

The architecture of the system is displayed in Figure 5. Here we can see three layers, the input, main pipeline and the final output. The input is a collection of one or more pdf documents belonging to a single court case. The text will be extracted from the documents and afterwards they are fed into the pipeline one by one.

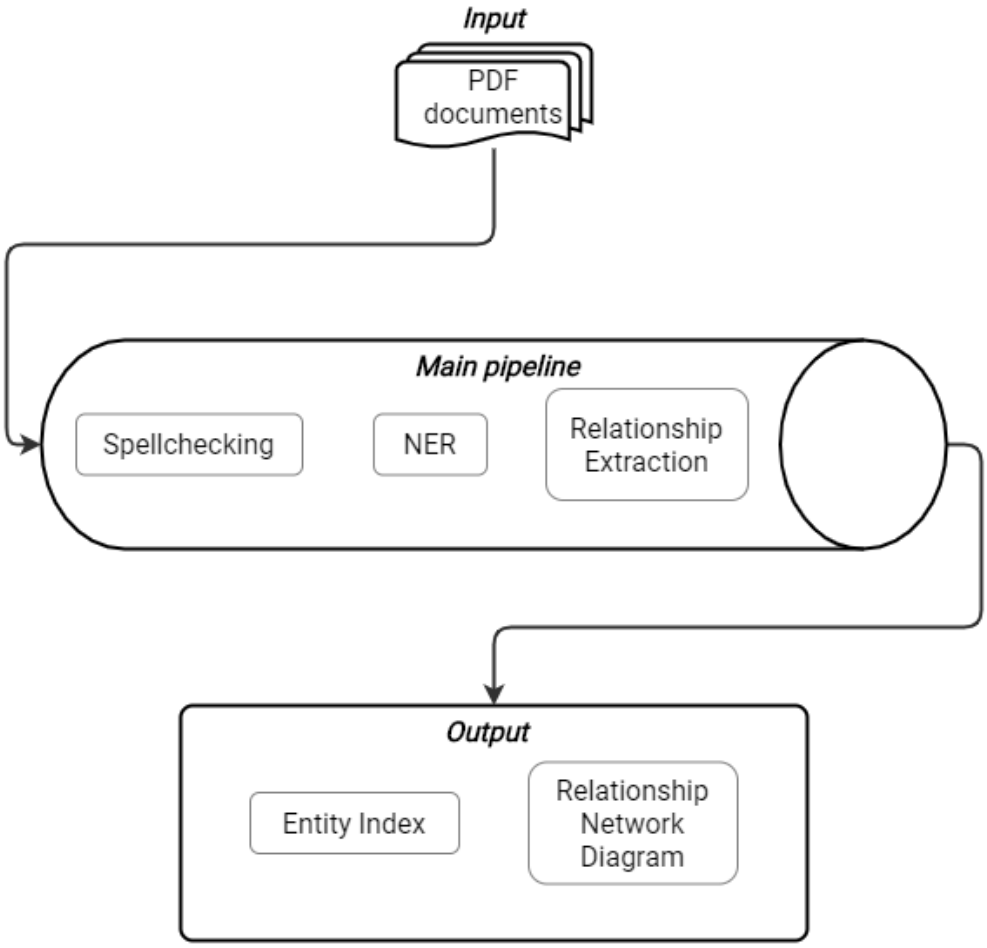

Figure 5: Overview of the integrated system

The pipeline contains the major processing components, spellchecking, NER and relationship extraction, which are discussed in the previous chapters. Each step in the pipeline is executed sequentially and produces some output that used by the next step. These different steps are loosely coupled, meaning that the subsequent steps do not depend on the implementation of earlier steps. This makes the design modular and easy to modify. For example, if the system were to use a different NER model, this step could easily be replaced without altering the subsequent relationship extraction step, as long as the same encoding for the entity categories is used.

Figure 6: Anonymised relationship network diagram for system output.

Finally when all the documents have passed through the pipeline, the results are combined to form the entity index and the relationship network diagram. The index is list of all unique detected NEs mapped to the documents in which they occur. The other output is the relationship network diagram, an anonymised example for system output is shown in Figure 6. This diagram contains a number of nodes corresponding to the unique NEs that were detected in the documents. The main central node is the subject of the documents and the other nodes depict NEs that are related to the subject.

7.2 Relationship diagram

This section describes the details about the relationship diagram visualisation. These details include how the data in the graph is determined from the earlier steps, and how to read the diagrams. The diagrams were developed using the following assumptions:

- Ambiguous names (with the same initials and lastname) can be resolved to the same real world person.
- All Named Entities have a direct (first degree) relation to the subject.

At first glance it might seem strange to assume this first point, especially since the dataset contains family related matters where this might occur often. However, the authors of the documents will take care of this disambiguation by not using the same initials with lastname reference for different real world persons.

The second assumption is similar to the one mentioned in section 6.2 and motivated by the fact that detailed expert reports were used as the data set. Of course all NEs from a case relate back to the subject in some way. For our purposes we assume that it is always a direct relation to the subject since the documents often describe directly what happens to the subject and by whom.

How to read the diagram

The visualisation diagram shown in Figure 6 contains all the found named entities and relations for an example case. The central node is always the subject of the case files. The nodes around it are other NEs that were detected and relate back to the subject in some way. The text in the node shows the NE as it occurred in the text. The colour of the nodes indicates the primary class: red for organisations, blue for locations, and green for persons. Recall that the person NEs have been further divided into subcategories in the relationship extraction component. The subcategory of the person nodes is indicated by the <*subcategory*> label under the name of the person.

Determining categories

Most of the NEs found in the documents occur more than once. The final category that is depicted in the diagram is based on a majority vote over all occurrences of a Named Entity. The category with the most predictions is

assigned to the node.

The central node of the diagram is is the subject of the documents and based on the NE with the most *Client* predictions. All other NEs which have Client as their most predicted category are not shown in the diagram. These are either wrong predictions or different forms of the subject's name, which would result in more uninformative nodes.

7.3 Experiments and Results

The relationship diagram figures shown in this chapter are all based on the same anonymised example case. This case is randomly selected from all average sized cases, hence this is what the average graphs will look like in terms of size and complexity. The system diagram that we see has many false positives for organisations due to the writer using a lot of (uncommon) abbreviations.

Another thing to note is that there are two outliers in the overall dataset which contain documents that are roughly ten times as large as the average. As a result these cases contain many more detected NEs than our example, although since many of them are repeated there are only about twice as many unique Named Entities in those documents. Consequently, the graphs resulting from those outliers also contain roughly twice as many nodes. On the other hand there are also a number of cases with just a single document, these produce graphs that are about half the size.

Case study

In order to find out how effective the diagram is with regards to research question 3, we performed a small case study with an expert of Justid. Together with the expert, we looked at what was good, what could be improved short term and what could be done in future work to improve effectiveness of the visualisation for providing a quick overview of the case.

Based on the case study with earlier versions of the diagram up to the current one, these were the main points identified by the expert that still remain:

- The diagram is too busy and complex to see what is going on quickly.

- The name format for persons is too inconsistent.
- There are too many wrong or uninformative nodes.

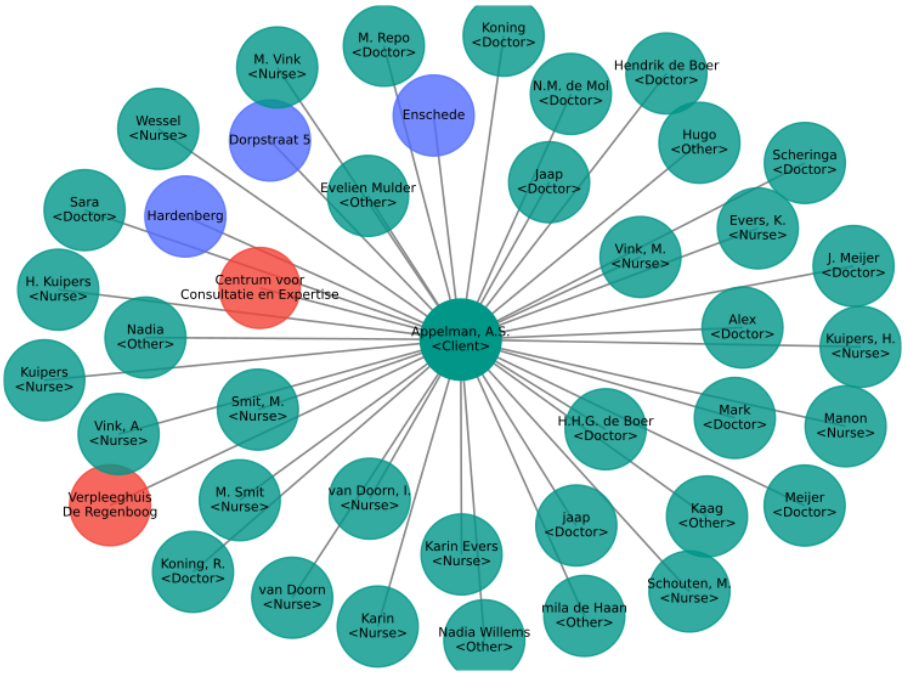

Figure 7: Anonymised relationship network diagram for ground truth annotations.

Figure 8: Anonymised relationship network diagram for ultimate goal.

Some short term adaptations based on the feedback have already been made. The colour coding scheme that was used to easily distinguish the categories helps with the overload of information, making it easier to identify the nodes. Initially all the person subcategories would also have a unique colour, however this resulted in too many colours to easily distinguish the categories. Instead we opted to display this under the person name with the angular braces to make it stand out. The colours in the diagram are also softer than before. Additionally some NEs were too long and caused them to overlap with other nodes. This has been fixed by displaying the words of these NEs on multiple lines, reducing the overlap for most NEs.

Goal diagram

There are still some problems with the current version regarding the points mentioned above. For this reason we have also manually created a goal diagram to strive for. This goal diagram is visually the same to make it easier to compare the current versions with. The contents of the goal diagram are altered to address all of the main points identified by the expert. The current version based on the ground truth annotations is shown in Figure 7 and the goal version is shown in Figure 8.

In the current version there are a lot of different mentions of the same real world people. We have applied a manual form of Coreference Resolution in order to reduce the number of nodes without losing information and also to make the name format more consistent. For this we use the previously mentioned assumption that ambiguous names can be resolved to the same real world person. All different mentions of persons are combined and they are included with the most informative name format. These are in order of precedence:

1. Full first name and last name (e.g. *Hendrik de Boer*)
2. Lastname and initials (e.g. *de Boer, H.*)
3. Only first or lastname (e.g. *Hendrik* or *de Boer*)

An example of this is for the nodes *Karin*, *Karin Evers* and *Evers, K.*. These are three forms of referencing the same real world person. In the goal diagram only the most informative version (*Karin Evers*) is included. Applying this method reduces the size of the graph considerably. The original diagram

which contains 42 nodes is reduced to 29 nodes in the goal diagram while not losing any information.

7.4 Discussion

The misinformation in the diagram is caused by compounding errors from the components in the pipeline and the original OCR layer in the documents. Each of these steps may introduce new errors and when each component depends on the output of the previous step, these errors propagate all the way to the final result. The majority vote to determine the category for a NE helps to correct some of these errors since the correct category will be predicted the most, however this only works for NEs that occur often and thus have more predictions.

There is also a limitation with regards to the relations that we see in the diagram. In chapter 6 we saw that relationships could only extracted for person NEs. For organisations this is fine, with domain knowledge it is often clear what the role of an organisation is, for example we can assume "Verpleeghuis De Regenboog" (eng. "Nursing home The Rainbow") housed the subject where they could receive extra aid. However, the role of locations is not so clear. In the example we can see different locations and it is not known whether the subject lives there, goes to one of the locations for consultation with a doctor or something else. In the future when more data is gathered, it might be possible to divide the locations into subcategories with the same method that was used for persons.

The final adjustment that can be done in the future is to keep a knowledge base of nurses doctors that are encountered in any document. Nurses and doctors often occur in documents across different case files, and sometimes only part of their name is revealed. It can be beneficial to keep a knowledge base with all these persons that are encountered and whenever there is a nurse or doctor that is only mentioned by part of their name, the full name can potentially be retrieved from this knowledge base which helps to make the diagram more complete and the naming format more consistent.

7.5 Conclusion

In this chapter we saw how the system was built and what the output looks like for the current setup. We conducted a case study with an expert

from Justid in order to answer research question 3: *How can the detected Named Entities and their relations be effectively represented in order to improve the workflow?* From the case study we can conclude that the current system output is not effective enough to do this due to the following problems. The main issues with the current relationship diagram are that there is too much information to quickly identify what is going on in the diagram, there is some misinformation in the diagram where NEs have the wrong category assigned to them, and the naming format is inconsistent. Based on the case study a goal diagram has been constructed to address the main problems with the current version.

The goal diagram is based on the ground truth annotations so the errors are not present, which reduces the size of the diagram. In order to achieve this for the graph generated by the system, the NER step has to be improved. As we can see from the example graph in Figure 6 there are a lot of abbreviations mistaken for organisations. As mentioned before in section 5.3 a good way to combat this problem is by creating a blacklist with abbreviations from the legal and medical domain that should not be detected as Named Entities.

The other big difference is that different references to the same real world persons have been removed systematically to make person names more consistent and remove duplicate information. This has been done manually for our goal diagram and has to be automated in the future. There is a specific field of research in NLP called Coreference Resolution that deals with this problem. When all the different references to unique real world persons are resolved, the same rules can be applied to have a consistent name format.

8 Conclusion

This thesis describes an exploratory study to determine what information can be extracted from scanned court case documents in order to improve the workflow. Various Natural Language Processing techniques were applied in a series of both successful and unsuccessful experiments. We have developed different components each with their own challenges that contribute to this goal, which were finally integrated into one system. First the documents were cleaned in the preprocessing step to reduce artefacts of the OCR and generally preparing the data for the subsequent components. The NER component was aimed at extraction all important locations, persons and organisations from the documents. Next, the role of each person with respect to the subject of the court case was determined. Finally, all the information gathered in previous components was combined to produce the output that is presented back to the user in the form of a entity index and a relationship network diagram.

Answering the Research Questions

Chapter 5 investigated research question 1: *How do modern transformer models compare to Conditional Random Field models on the task of Named Entity Recognition for the given dataset?*. Two different state-of-the-art architectures were examined, BERTje using the transformer architecture, and StanfordNER which uses a CRF architecture. Initially the performance for either model was not great. Based on the frequent errors by the models a number of modifications were made to improve the performance. In the end, BERTje which scored approximately the same on benchmarks performed considerably worse for our data. As a result StanfordNER, using the CRF architecture, was used as the NER model for the system. BERTje uses a model structure and training data very similar to the original BERT model and other variants of it. With regards to the research question we conclude that in our experiment the StanfordNER model was better than the BERT based model and it is likely that it will also be better than similar BERT based transformer models.

In chapter 6 the relationship extraction component of the system is described, which is aimed at answering research question 2: *What kind of useful relationships can be found between the detected Named Entities?* In this chapter we saw the initially planned approach, aimed at identifying relations between pairs of detected Named Entities. Unfortunately, after doing the annotations,

it appeared that the structure of the documents is not well suited for this and there were very few relations present that fit this pattern. Instead, the approach was altered to identify the relation between all the identified persons and the subject of the court case. So to answer the research questions, the system can detect whether a person Named Entity is the client, or family, a doctor, a nurse or a guardian of the client.

Finally chapter 7 set out to answer research question 3: *How can the detected Named Entities and their relations be effectively represented in order to improve the workflow?*. In short there are two ways to display information. An entity index which serves to provide a reference for users who already know what Named Entities they are looking for and points them to the right documents. The second way of representing the information is by showing a diagram containing all the detected Named Entities and their relations. This visualisation is targeted for users unfamiliar with the case and serves to provide a high level overview of the case at a glance. A case study with an expert was conducted to evaluate how effective this visualisation method is. Based on that some modifications have been made to the current version of the diagram, however this was not enough to make it an effective visualisation. We also showed an ultimate goal version of the diagram that addressed the feedback of the case study, and which serves as a proof of concept. In short to answer the research question, the relationship diagram is an effective way of displaying the detected Named Entities and their relations when the manual approach for the goal diagram is automated.

Overall many different NLP methods have been combined throughout the study in order to develop all the components. This was primarily an exploratory study to see what information could be extracted from the documents and how well this would assist with the workflow of people using the documents. We can conclude with respect to the main research question that this system is currently not good enough to support improve the workflow. It mainly serves as a proof of concept of what can be achieved. We expect that with more future development the system can be improved to effectively support the workflow of people who work with these documents.

Future Work

For the NER component, most errors were found with the organisations. The system incorporates a small blacklist of words that are ignored from the

predictions. The current version of the list only contains general Dutch words and abbreviations that were found in the most frequent errors of the models. The best way to further reduce the number of NER mistakes is by having domain experts extend this list based on the frequent errors. This can be done primarily for the words and abbreviations originating from the medical and legal domains.

Another way to improve the system is to maintain a knowledge base for certain person Named Entities as mentioned in section 7.4. This knowledge base can store information on person NEs that appear across different court cases. Later on, the system can use the information in the knowledge base to improve accuracy of the relationship extraction for nurses and doctors as well as making the naming format for those NEs in the visualisation more consistent and complete.

The main goal for future work should be aimed at implementing Coreference Resolution techniques. Currently a form of that technique is applied manually to create the goal relationship diagram. If this technique is implemented in an automated way, that would allow the system to produce an output relationship diagram that is much closer to the goal diagram.

9 Practical Observations

During the research conducted for this thesis not everything worked out as planned beforehand. We encountered some difficulties in the planned approaches and even had to completely redo the relationship extraction approach. Aside from the scientific contribution of this thesis and the practical contribution of the system, this chapter will describe some practical observations that can hopefully guide future researchers who work with similar conditions.

- The first observations result from working with real world data opposed to clean lab data. The data we used for this thesis was already being used for a different project within Justid, which lead us to assume it was easy to work with and we did not have to plan for the cleaning process. Additionally very often the courses at university present clean lab data that is ready to be used for the creation of prediction models or other goals without much preprocessing. However, in practice it turns out that this is not always the case. We have seen a number of complications throughout the thesis and many of them stem from the initial input to the system, where there are various artefacts from OCR and spellchecking that impact the performance of other components. To prevent having some of the issues we encountered, we recommend not to make assumptions about the data set and always do a thorough analysis of the data set beforehand.

- The second set of observations have to do with the privacy sensitive nature of the data. I had to sign a confidentiality contract allowing me to work with the data, however my supervisors did not have clearance to see it. This sometimes made the meetings inefficient because I could not clearly show the problems we had to deal with. Some advise that may help with this problem is to start communicating early about the structure and potential problems with the data through anonymised examples. This way everyone is on the same page and feedback will be more relevant and effective. Another thing that we encountered is that most of the state-of-the-art spellchecking solutions are web based. However due to the sensitive nature of the data, we could not use these web services for spellchecking. This is important to keep in mind when planning the approach to your research.

- The next observations have to do with the goals of industry and re-

search, which do not always align well. In general the end result is most important for the industry, while for research purposes the process of getting to said result is more important. When doing a research project for an external company, this can put additional workload on you in order to meet the demands of both parties. An example of this is for the NER step where based on the literature research the transformer model should have performed better than the alternative. After the initial results were known we had a CRF model that was already performing decently well, while the transformer model was not. For the company it makes sense to simply continue with the better model which has adequate results instead of trying to improve the weaker model. However, for research purposes we attempted several methods, such as doing the manual cleaning, in order to find out why the transformer model was so considerably worse in our case. It is recommended to schedule meetings regularly with both parties involved and attempt to establish a solid middle ground in order satisfy the wishes of both sides, without spending too much time on each part of your research.

- The final set of observations have to do with working from home. When we started the research, the pandemic just hit our country and everyone had to work from home. I visited the office of Justid a few times before for administrative purposes such as discussing the potential research and signing contracts, but did not know many colleagues and had not actually worked on location yet. It took quite some time to get used to the organisation, get familiar with the systems and environments and gain access to everything needed for the research. The contacts from Justid have been very helpful and responsive in getting ready, however it is always easier to get familiar with the organisation and get set up when you are physically there instead of doing this via emails. My advice therefore is to get to know the organisation and physically go there if at all possible. This will likely save time to get ready and start your research.

References

Cardellino, C., Teruel, M., Alemany, L. A., & Villata, S. (2017). A low-cost, high-coverage legal named entity recognizer, classifier and linker. In *Proceedings of the 16th edition of the international conference on artificial intelligence and law* (p. 9–18). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3086512.3086514 doi: 10.1145/3086512.3086514

Carreras, X., Màrquez, L., & Padró, L. (2002). Named entity extraction using adaboost. In *Coling-02: The 6th conference on natural language learning 2002 (conll-2002).*

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019, June). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/N19-1423 doi: 10.18653/v1/N19-1423

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., & Nissim, M. (2019). Bertje: A dutch BERT model. *CoRR*, *abs/1912.09582*. Retrieved from http://arxiv.org/abs/1912.09582

Dozier, C., Kondadadi, R., Light, M., Vachher, A., Veeramachaneni, S., & Wudali, R. (2010). Named entity recognition and resolution in legal text. In E. Francesconi, S. Montemagni, W. Peters, & D. Tiscornia (Eds.), *Semantic processing of legal texts: Where the language of law meets the law of language* (pp. 27–43). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-12837-0_2 doi: 10.1007/978-3-642-12837-0_2

Finkel, J. R., Grenager, T., & Manning, C. D. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)* (pp. 363–370).

Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Ó Séaghdha, D., Padó, S., & Szpakowicz, S. (2010, July). SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 33–38). Uppsala, Sweden: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/S10-1006

Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing (2nd edition)*. USA: Prentice-Hall, Inc.

Levenshtein, V. I. (1966, February). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, *10*, 707.

Pires, T., Schlinger, E., & Garrette, D. (2019, July). How multilingual is multilingual BERT? In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4996–5001). Florence, Italy: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/P19-1493 doi: 10.18653/v1/P19-1493

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020). Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th annual meeting of the association for computational linguistics: System demonstrations.* Retrieved from https://nlp.stanford.edu/pubs/qi2020stanza.pdf

Ramshaw, L. A., & Marcus, M. P. (1999). Text chunking using transformation-based learning. In *Natural language processing using very large corpora* (pp. 157–176). Springer.

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., & Tsujii, J. (2012, April). brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the demonstrations session at EACL 2012.* Avignon, France: Association for Computational Linguistics.

Tjong Kim Sang, E. F. (2002). Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of conll-2002* (pp. 155–158). Taipei, Taiwan.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In I. Guyon et al. (Eds.), *Advances in neural information processing systems* (Vol. 30, pp. 5998–6008). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Zhang, Y., Zhong, V., Chen, D., Angeli, G., & Manning, C. D. (2017). Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 conference on empirical methods in natural language processing (emnlp 2017)* (pp. 35–45). Retrieved from https://nlp.stanford.edu/pubs/zhang2017tacred.pdf

Appendix A: Extra NER results

Table 14: Micro averaged impact of cleaning documents for the BERTje NER model.

	Recall	Precision	F1
Uncleaned	0.29	0.21	0.24
Cleaned	0.35	0.24	0.29

Table 15: Detailed NER result for different postal code correction methods.

	Precision				Recall			
	Per	Loc	Org	Other	Per	Loc	Org	Other
BERTje								
standard	30.3%	33.1%	3.1%	98.8%	37.4%	44.2%	8.5%	98.0%
removal	30.3%	34.5%	3.1%	98.8%	37.4%	44.2%	8.5%	98.0%
strip	30.3%	37.3%	3.1%	98.9%	37.4%	49.9%	8.5%	98.0%
Stanford								
standard	71.3%	57.6%	30.7%	99.5%	74.7%	69.8%	50.3%	99.1%
removal	71.4%	59.8%	31.4%	99.5%	74.7%	69.8%	50.3%	99.1%
strip	71.3%	61.0%	30.7%	99.5%	74.7%	74.0%	50.3%	99.1%