Department of Informatics

Technical University Munich

Bachelor's Thesis in Informatics

Anonymization of German Legal Texts

Tom Schamberger

Department of Informatics

Technical University Munich

Bachelor's Thesis in Informatics

Anonymization of German Legal Texts

Anonymisierung von deutschen Rechtstexten

Supervisor:	Prof. Dr. rer.nat. Florian Matthes
Advisor:	M.Sc. Ingo Glaser
Submission date:	15th of November 2019

Acknowledgements

I confirm that this Bachelor's thesis is my own work and I have documented all sources and material used.

Munich, 6th of November 2019

Tom Schamberger

Zusammenfassung

Im Rechtsbereich werden viele rechtliche Dokumente wie Gerichtsentscheidungen und Verträge regelmäßig anonymisiert. Für diesen Vorgang müssen Textstellen mit hoher Sensitivität identifiziert und neutralisiert werden, um vertrauliche Informationen vor Dritten zu schützen. In der Regel wird dieser Vorgang manuell von geschulten Mitarbeitern durchgeführt. Daher wird die Anonymisierung im Allgemeinen als teurer und ineffizienter Prozess angesehen.

In dieser Arbeit wird ein Modell-basierter Ansatz zur automatischen Identifizierung sensitiven Textstellen in deutschen Gerichtsentscheidungen entwickelt. Hierfür werden verschiedene Architekturen tiefer neuronaler Netze basierend auf allgemein vortrainierten Kontext-basierten Einbettungen (engl. "contextual embeddings") sowie spezifisch trainierten Wort-basierten Einbettungen (engl. "word embeddings") evaluiert. Aufgrund der geringen Verfügbarkeit nicht anonymisierter Datensätze werden die Modelle ausschließlich auf anonymisierten Daten trainiert. Um diese Einschränkung zu überwinden, wurden die Architekturen der neuronalen Netzwerke so entworfen, dass die Klassifizierung von Textelementen hauptsächlich vom umgebenden Kontext abhängt. Darüber hinaus wurde ein regelbasierter Algorithmus entwickelt, um anonymisierte Textstellen in juristischen Dokumenten zu kennzeichnen. Schließlich wurden die Modelle und Algorithmen anhand manuell umgeschriebener Rechtsdokumente evaluiert.

Abstract

In the legal domain, many legal documents such as court decisions and contracts are regularly anonymized. This process requires text sequences with high sensitivity to be identified and neutralized to secure sensitive information from third parties. Usually, this process is performed manually by trained employees. Therefore, anonymization is generally considered an expensive and inefficient process.

This thesis proposes a machine learning approach for the automatic identification of sensitive text elements in German legal court decisions and provides an implementation. For this task, different deep neural network architectures based on generally pre-trained contextual embeddings as well as trained word embeddings are evaluated. Because of the lack of non-anonymized data sets, the machine learning models are solely trained on anonymized data. To overcome this limitation, the neural network architectures have been designed in such a way that classification of text elements mainly depends on the surrounding context referred to as "contextual sensitivity classification". Furthermore, a rule-based algorithm has been developed in order to label anonymization placeholders in anonymized legal documents. Finally, the models and algorithms have been evaluated using manually rewritten legal documents.

Contents

Glossary

BERT	Bidirectional Encoder Representations from Transformers
GloVe	Global Vectors for Word Representation
MLM	Masked Language Model
MSL	Maximum sequence length
NER	Named entity recognition
NLP	Natural Language Processing
NSP	Next Sentence Prediction
Python	Interpreted, high level programming language
Regex	Regular expression
Regular expression	Sequence of characters that define a search pattern
Sensitivity	Measure determining potential harmfulness in case of public disclosure
Tensorflow	Open source platform for machine learning
Text Element	Smallest text fraction able to contain a reference or a placeholder
TFRecord	Binary file format for data sets in Tensorflow
Tokenization	Process of splitting text into tokens

List of Figures

List of Tables

1. Introduction

1.1. Motivation

Legal documents such as contracts or court decisions are regularly being anonymized, in order to be published or handed out to third parties. Usually, such documents are manually anonymized by skilled employees. For this reason, many organizations consider anonymization an expensive and inefficient process. In recent years, this has led to a scarcity of publicly available legal data sets. But court decisions in particular represent vital base material for legal professions, academic researchers, journalists and private companies [Op17]. Especially lawyers and law firms require those documents for case research, because cases are usually evaluated by means of related cases. Furthermore, the lack of publicly available data sets inhibits legal innovation utilizing machine learning or big data techniques. According to a decision of the German Federal Administrative Court (BVerwG), German courts are obliged to publish completed verdicts [E1]. This makes anonymization an inevitable task for most German courts.

This thesis proposes a machine learning approach to automate the identification of sensitive passages in German legal documents. The proposed approaches are intended to support legal employees with the anonymization of legal texts and help the legal industry to develop fully-autonomous anonymization solutions.

However, the automation of anonymization tasks usually requires both anonymized and non-anonymized data. Due to data protection policies, data sets including non-anonymized data are seldom publicly available. Placeholders used to label anonymized named entities in legal text corpora may be manually replaced with random representations of the corresponding named entity type.

However, this process is complex, time-consuming and limits the size of trainable data. In this thesis, models are trained solely on the surrounding context of anonymized text passages, so that neither manually rewritten nor non-anonymized legal documents are required.

1.2. Research Questions

In this thesis, three objectives have been investigated:

1. How can placeholders be detected in anonymized legal documents?

The legal documents examined in this theses have been anonymized using reference removal (see 2.1). This means that references have been replaced with special placeholder. In this work, this property is used to train machine learning models on the context of those placeholder. However, the way how placeholders are expressed in current German court decision is highly inconsistent both within and between documents. Also, the applied replacement method varies heavily. Because of this inconsistency, the detection of anonymization placeholders represents an important part of this work.

2. How can machine learning approaches be used to automate anonymization using only anonymized data?

The second objective investigates, how German legal documents having been anonymized using labeled placeholders can be used to train machine learning models to distinguish between sensitive and in-sensitive text elements. In this thesis, this method, referred to as "contextual sensitivity classification", is evaluated using different machine learning architectures that have been inspired by state-of-the-art NLP deep learning architectures. However, due to the special requirements of this approach, new architectures are additionally introduced. Thereby, pre-trained embeddings have been used to reduce the difference between the training and the test data, referred to as "validation-test gap", further discussed in section 7.2. Furthermore, state-of-the-art pre-trained word- and contextual embeddings used in this work provide additional general language knowledge, so that less legal documents are required for model training.

3. Is the textual context of text elements enough information to predict sensitivity in German legal texts?

The third objective aims to answer if the context of text element combined with generally [1] pre-trained embeddings provides enough information to predict the sensitivity of text elements of German legal texts. For that purpose, the prosed deep learning architectures are evaluated on manually rewritten German legal documents using three evaluation methods: At first, the ability to distinguish randomly chosen text passages in respect to their sensitivity is investigated. Furthermore, a NER model is used to pre-select named entities from the legal text corpus in order to study how this influences classification performance of proposed models. Finally, the models are directly applied to the test corpus and the performance is analyzed.

1.3. Structure of this thesis

First of all, fundamental principles of this thesis are explained in chapter 2. In chapter 3, related work is explained. Then, the scientific approach of this work is explained in chapter 4.

In chapter 5, the methodologies are described that have been used to obtain the results. In section 5.2, the legal text corpus and the associated assumptions are presented. Then, insights on how legal documents were prepared for training are given in section 5.3. Next, the model architecture, implementation and parametrization is described in section 5.4. In section 5.5, the evaluation process is explained.

The evaluation results are presented in chapter 6. In chapter 7, the results are further discussed. Then, a conclusion is drawn in chapter 8. Finally, suggestions for future work are made in chapter 9.

[1] In this context, general means that the embedding has been trained on a non-specialised corpus such as Wikipedia or OpenBooks

2. Basic knowledge

In this chapter, fundamental concepts and methods are explained, on which the approach presented in this thesis is based.

2.1. Anonymization

Anonymization is defined as the task of identifying and neutralizing sensitive references within a given document or a set of documents [Me06]. Sensitivity is a binary measure determining whether or not a particular reference, if publicly disclosed, might potentially cause harm or engender undesirable personal or legal repercussions [Me06].

Therefore, in order to anonymize documents, sensitive information has to be identified first. According to the definition of sensitivity, this information must contain at least one direct reference to an object or a juristic person outside the context of the document. In this thesis, those text fractions are referred to as references. As a second step, identified references have to be neutralized. In general, there are three possibilities of neutralization [Me06]:

- Removal: Replacement with a placeholder
- Categorization: Replacement with a label revealing type information
- Pseudonymization: Replacement with a random variant of equal type

In order to preserve the meaning of the text, it is necessary to replace references referring to the same real object or person in such a way that the connection within the text stays intact. However, this issue is outside the scope of this thesis.

The most challenging aspect of anonymization is the identification step. Therefore, in this work, the anonymization problem is reduced to a text sequence classification task referred to as contextual sensitivity prediction. Thereby, it is not sufficient to only detect references, because in legal documents such as court decisions, many references are insensitive. Instead, the sensitivity of references in legal documents mainly depends on the textual context. For instance, court names reference real-world objects, but they are insensitive and contain useful information for reviewers. On the other hand, references to expert witnesses must not be exposed as they are highly sensitive. The following example (excerpt of an anonymized court decision) demonstrates this fact:

Example 2.1.1 *"(...) Dämmwerte zwingend vorschreiben würden (vgl. BayObLG NZM 2002, 75). Dies hat der Sachverständige ... in seinem Ergänzungsgutachten vom 21.9.2006 (dort S. 28 ff, S. 33) nachvollziehbar klargestellt."* [2]

In this example, the expert witness (German: "Sachverständige") is anonymized, while the court name of the Bavarian higher state court (German: "BayObLG") is revealed. Because the anonymization of this text only depends on the context of references, the paragraph can be pseudonymized using a random instance from a list of names:

Example 2.1.2 *"(...) Dämmwerte zwingend vorschreiben würden (vgl. BayObLG NZM 2002, 75). Dies hat der Sachverständige Rainer Kurt in seinem Ergänzungsgutachten vom 21.9.2006 (dort S. 28 ff, S. 33) nachvollziehbar klargestellt."*

2.2. Sequence Classification

The anonymization problem described in section 2.1 can be reduced to a sequence classification problem referred to as contextual sensitivity classification. Sequence classification aims to predict labels for a sequence of input vectors over space or time. In case of contextual sensitivity classification, the sequence of input vectors represents a sequence of tokens within text documents. Thereby, each token is classified as sensitive or insensitive depending on its textual context.

[2] LG München - 1 T 15543-05 - juris Rechtsdatenbank

2.2.1. Tokenization

The process of splitting a text document into tokens is referred to as tokenization. The aim of this process is to transform text into a sequence of samples drawn from vocabulary of finite size.

In this work, two different tokenizers are being used: The BERT WordPiece tokenizer used internally by BERT [De19] (see 2.2.2.1) and the SentencePiece tokenizer used for the specifically trained GloVe embeddings. Both tokenizers split words into subwords in order to prevent out-of-vocab issues. BERT's WordPiece tokenizer simply splits text on white-spaces and maps words onto one or multiple tokens. Thereby, less frequent and long words are split into more tokens than frequent words.

The SentencePiece tokenizer [3] is an unsupervised text tokenizer used to generate the vocabulary of predetermined size. It is trained on raw sentences and supports either byte-pair-encoding (BPE) [SHB16] or the unigram language model [Ku18]. The latter is used in this work. The SentencePiece tokenizer is trained on raw sentences, which means that no pre-tokenization such as white-space splitting is necessary. Thereby, it splits words into subwords like the WordPiece tokenizer being used by BERT models. This enables a finite vocabulary without out-of-vocab incidences. Furthermore, whitespace is treated like a basic symbol such that no information about spaces between characters is being lost.

2.2.2. Embeddings

Embeddings are used to transform tokens represented by integer ids into vectors that encode their syntactical and semantical meaning. There are both contextual and word embeddings. Contextual embeddings aim to encode the meaning of the token in relation to its textual context. Therefore, the whole input sequence is necessary to produce those embeddings. Word embeddings on the other hand aim to encode the universal meaning of tokens independent of their context. Thereby, the token ids can be directly mapped to embedding vectors using a lookup table.

[3] https://github.com/google/sentencepiece

2.2.2.1. BERT

The Bidirectional Encoder Representations from Transformers (BERT) [De19] is a state-of-the-art masked language model (masked LM) utilizing the Transformer architecture [Va17] to produce contextual embeddings. This architecture makes use of attention layers, instead of recurrent neural networks (RNNs) or convolutional layers, combined with an encoder-decoder architectures. The pre-trained multilingual model used in this thesis [4] has been trained on multinational Wikipedia using unsupervised learning methods such as masked language modeling (MLM) and next sentence prediction (NSP). The MLM aims to predict randomly masked input tokens depending on its surrounding context. The NSP is a sequence classification task and is used to grasp the meaning of a whole token sequence.

2.2.2.2. GloVe

The Global Vectors for Word Representation (GloVe) [PSM14] is a method for unsupervised learning of vector space representations of words. It focuses on the capturing of fine-grained semantic and syntactic meaning a word and encode this meaning as a vector in finite dimensional space. Those vectors are trained using the co-occurrence statistics of the corpus. In this thesis, pre-trained GloVe representations [5], trained on German Wikipedia, as well as representations, trained on the legal document corpus, are used. Instead of contextual word representations, the GloVe representations are context-independent.

[4] https://tfhub.dev/google/bert_multi_cased_L-12_H-768_A-12/1
[5] https://deepset.ai/german-word-embeddings

3. Related work

In recent years, most anonymization systems, such as [SKU14], [Tv04] and [Sw07], have been developed for the biomedical domain. These systems aim to remove the private health information (PHI) from clinical records [ULS07] and are usually composed of the following modules [Di16]:

1. Pre-processing: The required features are extracted.
2. Named Entity Recognition: References (such as patient names) are detected.
3. Replacement: The detected references are neutralized.

Thereby, named entity recognition (NER) represents the most substantial module. NER is defined as the task of identifying named entities (synonymous to references in the anonymization context [Me06]) in running text [Be14]. This task is tackled using either rule-based or model-based systems.

Classical anonymization approaches, such as [Tv04] and [Sw07], make use of rule-based named entity recognition systems that rely on dictionary lookups and regular expressions to classify individual words. On one hand, rule-based methods usually suffer from weak robustness against rare occurences and outliers such as spelling mistakes as well as words that lie outside of the vocabulary, because manually defined rules usually are too simple to match the high variety of different named entity representations. On the other hand, less data is necessary compared to model-based NER systems [Di16].

Modern approaches make use of model-based named entity recognition for anonymization [Di16]. Current model-based NER systems use recurrent neural networks (RNNs) like stacked Long Short-Term Memory (LSTM) RNNs [HS97] in combination with conditional random fields (CRFs), in order to classify tokens in the IOB (Inside, Outside, Beginning) tagging scheme [La16]. This tagging scheme is used, since named entities may span over multiple tokens.

Thereby, the CRF guarantees compliance with the scheme guidelines. Due to the small amounts of publicly available NER training data sets, state-of-the-art NER architectures make use of pre-trained word embeddings such as GloVe [PSM14] [La16].

Some anonymization approaches, such as [DMB16], additionally use co-reference resolution after the NER step, in order to resolve dependencies of identified references in texts. This may lead to more consistent classification results [Di16].

Applied to the legal domain, the downside of all related methods presented is that all detected references are considered to be sensitive. As a result, insensitive information is unnecessarily neutralized as long as it contains any reference. In legal documents, entities such as dates and locations must also be considered references, because they may reveal sensitive information if combined with additional information from the document. Because many dates and locations are insensitive but yet essential to understand the meaning of the text, the above methods would unnecessarily discard vital information. Furthermore, model-based NER systems require large non-anonymized data sets to be trained. Currently, those data sets are not publicly available for the German legal domain.

4. Research Methodology

An overview over the applied research methodology based on [Fr17] is given in figure 4.1.

The German legal market including lawyers, law firms, legal departments and courts as well as legal data scientists and legal technology (legal tech) companies represents the environment of this work. All mentioned actors desire a solution that automates the anonymization of German legal documents in order to make the publishing process both more efficient and more common. The knowledge base consists of the foundations as well as the applied methodologies, on which this work is based. The foundations of this work such as pre-trained masked language model BERT and GloVe word embeddings are described in chapter 2. The methodologies that have inspired this approach are presented in chapter 3. The artifact of this research is the proposed placeholder detection algorithm and the model architecture as well as its implementation. Those artifacts are used to assess the objectives presented in section 1.2.

4.1. Iterative Approach

In this work, an iterative approach has been used. Each cycle, the following steps have been taken:

1. Literature review
2. Design phase
3. Implementation phase
4. Application phase
5. Validation phase

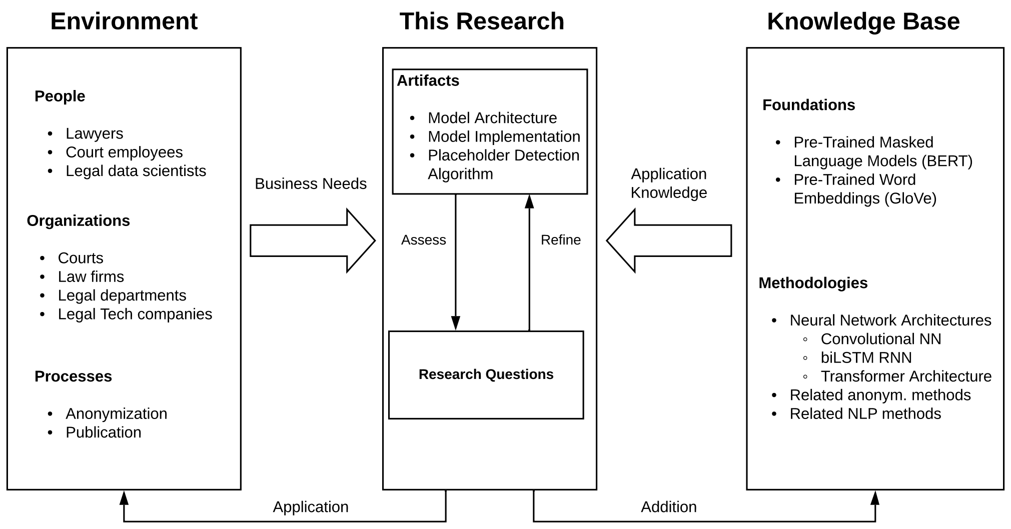

Figure 4.1.: Overview: Research Methodology

In the first cycle, more general literature research on the subject of anonymization and natural language processing (NLP) has been done. The artifacts of this research are the basic knowledge and related work chapters of this thesis. During the first design phase, the document fetching, document preprocessing and model evaluation pipelines have been designed using UML diagrams, partly included in this thesis. These pipelines are described in the methodology chapter (see 5). In the first implementation phase, those pipelines have implemented in Python. In the first application phase, the pipelines have been run and the resulting training and validation corpora have been compressed and stored. Finally, the pipelines have been manually validated on a sample basis and using unit tests.

In the following cycles, more specific literature research on different rule-based and machine learning approaches has been done. This step has been followed by a design phase, where the conceptual architectures and algorithms such as machine learning models or rule-based text processing algorithms have been modeled using custom and UML diagrams, partly included in this thesis. In the implementation phase, the models have been implemented using Python and Tensorflow (see 5). During application phase, the algorithms and models have been applied/trained on the text corpus. Finally, in the validation phase, the models and algorithms have been evaluated using the test data set. The results have been documented in the results chapter of this thesis.

5. Methodology

In this chapter, the specific procedures and techniques as well as implementation details are described.

5.1. Overview

In order to train and evaluate the deep learning model, legal documents had to be fetched from online sources. Thereafter, raw text and meta data has been extracted from fetched HTML documents (ETL). This raw text corpus consisting of 1500 German legal documents, further described in section 5.2, has been split into a training and a test corpus. Using the paragraph extraction, placeholder detection and tokenization steps, further described in section 5.3, the training corpus has been transformed into a data set of labeled token ids. During model training, discussed in section 5.4, the deep learning models have been optimized on the training data using specialized hardware. The resulting models have been evaluated using the test data set, which has been pre-processed and manually transformed in order to resemble actual non-anonymized legal documents. This process is further described in section 5.5.

An overview over the model training process is illustrated in an UML activity diagram in figure 5.1.

The entire process has been implemented in Python 3.7 due to its support for major machine learning libraries like Tensorflow. The following frameworks and libraries have been used:

- Numpy 1.17.0 [6]

[6] https://numpy.org

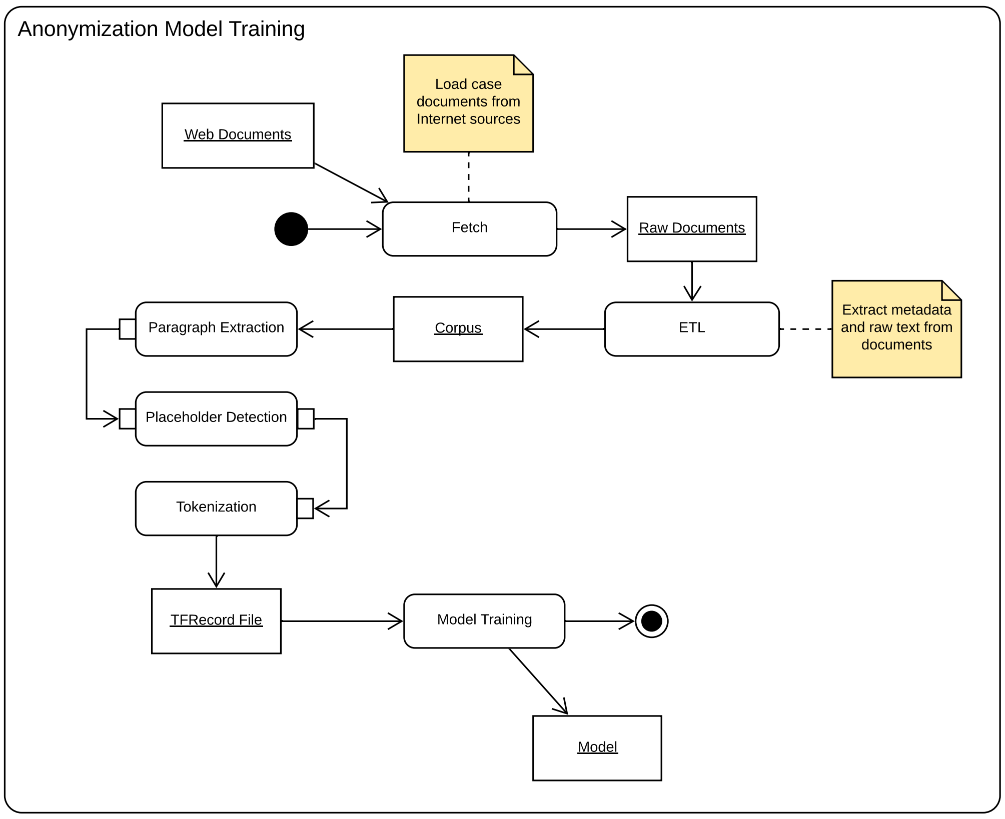

Figure 5.1.: Model Training Overview (UML 2.0 Activity)

- Tensorflow 1.14.0 [7]

Tensorflow is the most widely used and production-ready open source machine learning platform and provides its main interface in the Python language. NumPy represents a numeric library for vector and matrix computation in Python.

5.2. Legal Text Corpus

The data set used for training and validation consists of 1400 German anonymized court decisions of the state court in Munich (LG Munich) that have been published in recent years. [8] Most of the documents contain meta information such as case ID, dates and more. The meta information has been kept in separate JSONL files and documents were identifies using the case ID. In the legal text corpus, each anonymized reference has been removed during anonymization and replaced with a placeholder.

The following assumptions are made about the legal documents used for training:

1. All anonymized references have been neutralized using placeholders without further modification
2. Placeholders are easily [9] distinguishable from other text fractions
3. References have been replaced in such a way that the meaning of the text is retained
4. All documents have been anonymized using consistent and legally defined rules

Even though, the listed assumptions cannot be verified, the last assumption seems to be reasonable, because the documents have been released by the same court. The third point can be assumed because the documents have been published such that reviewers understand the legal meaning of cases. The second

[7] https://www.tensorflow.org
[8] Source: https://www.gesetze-bayern.de
[9] In this context, "easily" means that placeholders can be detected by simple, rule-based algorithms with high accuracy

assumption seems reasonable, since there are limited ways to express placeholders in text such that humans recognize them without clarification. In chapter 6, this assumption is confirmed by evaluation of the rule-based placeholder detection algorithm. The list of patterns for anonymization placeholders, which has been used in this work, can be found in the appendix A.1.

After the documents have been fetched, they were randomly split into a training corpus consisting of 1220 documents and a test corpus consisting of 180 documents, stored as compressed TAR files of raw text data. However, because the test corpus has to be manually transformed into the test data set in order to guarantee applicable and realistic results (see section 5.5.1), only a fraction of the raw test data has been used for evaluation.

The following table summarizes the most important information about the pre-processed training corpus:

Document count	1.220
Text element count	4.181.266
Anonymized element count	33.779
BERT WordPiece average tokens per element	1.8
SentencePiece average tokens per element	1.4

Table 5.1.: Information summary of the pre-processed training corpus

5.3. Pre-Processing

The pre-processing of the raw text corpus consists of three rule-based algorithms.

1. In the paragraph extraction step, documents is transformed into paragraphs of text without structure
2. In the placeholder detection step, anonymization placeholders are unified and labeled
3. In the tokenization step, the text is split and mapped to sequences of labeled token ids

For each algorithm, unit tests were implemented to ensure correctness. Furthermore, due to the importance of the placeholder detection step, an additional validation process has been developed.

5.3.1. Paragraph Extraction

The paragraph extraction algorithm, takes raw text sequences and produces paragraphs. In this thesis, paragraphs are defined as fractions of text, which contain independent statements. Accordingly, individual elements of a statement enumeration and separated text sections represent valid paragraph instances. This is reasonable, since legal texts such as court decisions are already formatted into enumerated paragraphs satisfying this condition.

Thereby, paragraph extraction are necessary, because non-recurrent deep learning architectures such as BERT only support text sequences of limited size. This introduces the data fragmentation problem [Da19], which causes tokenized sentences to loose their meaning, if they are split into multiple fractions. This problem is even amplified by the fact that most modern pre-trained language models use word-piece tokenizers, which split individual words into multiple tokens. This means that naive splitting of words may result in fragmented word parts on the edges of sequences. However, this problem can be minimized by splitting the text into paragraphs first and combining the paragraphs into sequences of fitting token length.

Paragraph extraction consists of the following steps:

1. Combine sentences connected by hyphenated word parts
2. Concatenate consecutive lines without separation (e.g. blank lines, enumerations)
3. Remove structural data (e.g. bullet points, enumerations)

5.3.2. Placeholder Detection

The placeholder detection is a rule-based classification algorithm, which takes paragraphs of anonymized legal documents and labels the anonymization placeholders within the paragraph.

At first, the paragraphs are split into text elements. In this thesis, text elements refer to the smallest text fraction, which is able to contain an entire reference or placeholder. Therefore, sensitive text passages can be neutralized by simply removing the respective text elements. While text elements may span across multiple whitespaces, e.g. in the case of dates or full names, the element splitting has been simplified to a white-space split. On one hand, full names or dates may be fragmented this way, so that each fragment have to be classified individually. On the other hand, this effect can be easily handled by the deep learning model, while no additional complicated algorithm is necessary.

The text elements are classified using a sliding window of 3 consecutive text elements. Each triplet of text elements consists of a predecessor, a anonymization candidate and a successor. Two different types of placeholders are distinguished using regular expressions: Obvious and potential placeholders (for examples see appendix A.1). Obvious placeholders meet strong criteria and represent placeholders that are specially marked by the author, e.g. '"E."'. Potential placeholders may be interpreted as placeholders if viewed outside the context, but may alternatively possess one of the following meanings:

- Omission within cites (e.g. testimonies)
- Abbreviation (e.g. "i.d.R.")
- Reference to pages or appendices (e.g. "siehe Anhang A 34")
- Reference to laws (e.g. "§ 8 Abs. 3 Ziffer 2 UWG")

The detected placeholders are labeled and replaced by a special symbol such that placeholders cannot be fragmented during tokenization.

5.3.2.1. Validation

Since the correctness of the deep learning model is based on the correctness of the placeholder detection, it is validated by the pre-processed validation data set, further discussed in section 5.5. In this data set, anonymization placeholders have been manually replaced with random instances of the anonymized type. For instance, the phrase "Herr Dr. A. G." might have been replaced with "Herr Dr. Arnold Griesmann" or any other name with the same initials.

This allows the validation of the placeholder detection by feeding the original anonymized text into the classifier and checking if the placeholder candidate has been manually replaced or not.

The results of the placeholder detection validation are presented in section 6.1.

5.3.3. Tokenization

After the placeholders have been labeled and replaced by special tokens, resulting paragraphs are tokenized and joined until the combined size would exceed the maximum sequence length (MSL). Very long sequences beyond the MSL are recursively split using a simple rule-based heuristic, trying to prevent sentence fragmentation. Due to the rarity of long paragraphs (6 occurrences in the training corpus), this algorithm has only been validated using simple unit tests. Three different tokenizers have been used: The WordPiece tokenizer for BERT embeddings [De19], the SentencePiece [10] and the Stanford [11] tokenizers for GloVe embeddings [PSM14].

For Glove embedded tokens, two tokenizers have been used in order to make use of the pre-trained word embeddings [12], which were pre-trained on unspecialised German Wikipedia using case-insensitive words generated by the Stanford tokenizer, and to additionally take advantage of embeddings, which have been trained directly on the specialized legal text corpus. So, each text element is mapped to a tuple of one "word token" (alphanumeric only) and one or multiple "word-piece tokens".

Finally, the resulting tokens are translated into ids and exported to a TFRecord file [13] with corresponding sensitivity labels (positive for placeholders, negative for non-placeholders), in order to be used for model training.

[10] https://github.com/google/sentencepiece
[11] https://nlp.stanford.edu/software/tokenizer.shtml
[12] https://deepset.ai/german-word-embeddings
[13] https://www.tensorflow.org/tutorials/load_data/tfrecord

5.4. Model Training

In this section the model architectures and training details are explained.

5.4.1. Model Architectures

Two embedding approaches were evaluated: Contextual pre-trained BERT embeddings and pre-trained GloVe word embeddings. On top of these embeddings, different layer architectures were tested.

5.4.1.1. Leave-Out Layers

In order to break the direct dependency between text elements and their classification (see 2.2.2.1), a special layer architecture has been implemented that makes predictions solely context-dependent by preventing information flow from input text elements to respective outputs. Two different variants are being evaluated: The leave-out convolutional layer (LOConv) based on a convolutional layer and a leave-out bidirectional RNN layer (LOBiRNN) based on a forward and backward directional LSTM cells [HS97]. Both variants yield exactly one output vector per text element, represented by the first token of the each element. One major disadvantage of both variants is that both architectures are restricted to singular use and cannot be stacked in order to preserve the desired property that the information flow between input text elements and respective outputs is prevented.

LOConv: For each output, each filter of the convolutional layer has been extended by a zero matrix such that the forced zero values of the filter correspond to the tokens of the respective text element. Because text elements consist of different numbers of tokens, this extension of the trained convolutional filters is adapted to each text element individually. The LOConv architecture is visualized in figure 5.2.

LOBiRNN: Multiple forward as well as backward LSTM [HS97] RNNs are stacked on top of each other such that forward layers cannot access outputs from backward layers and vice versa. The output of each element consists of

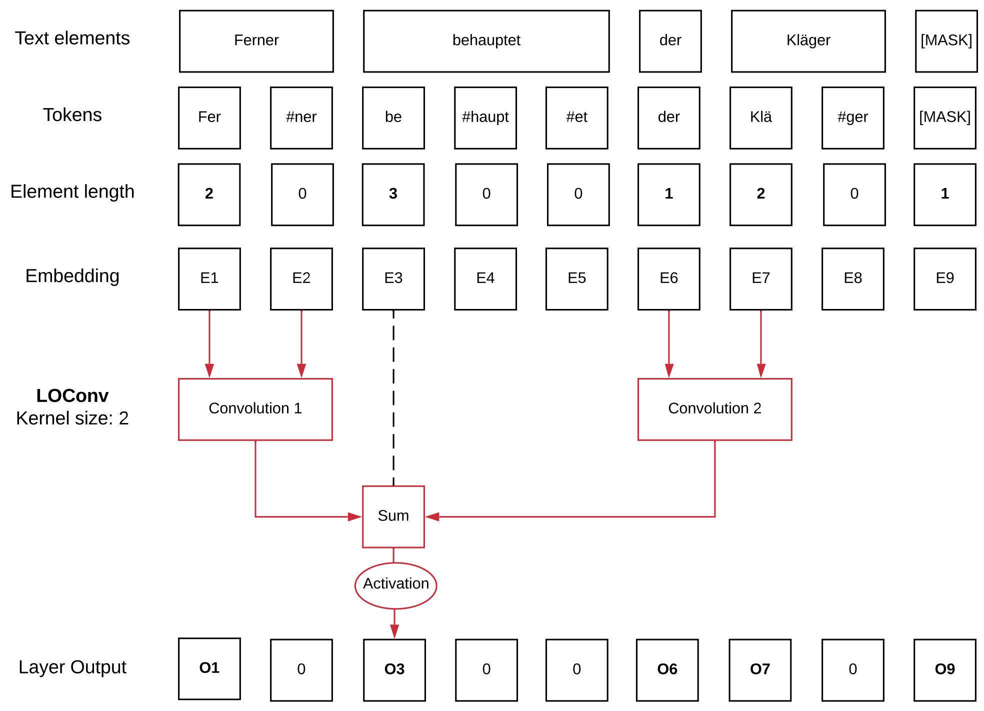

Figure 5.2.: Leave-Out Convolutional Layer (LOConv)

the final forward LSTM cell of the first token before the element, concatenated to the final backward LSTM cell of the first token after the element. The concatenated outputs are combined using a dense layer. The LOBiRNN architecture is visualized in figure 5.3.

5.4.1.2. BERT Embeddings

Because the legal data corpus consists of anonymized documents, input vectors contain placeholders during training that are not present during evaluation. In order to infer useful vector representations for those placeholders, a pre-trained masked language model is used. Masked language models have been trained on large text corpora such as Wikipedia by inference of masked input tokens. Thereby, the resulting embeddings for masked tokens are solely based on the context of those tokens. Therefore, masked language models aim to output embeddings for masked input sequences that resemble embeddings for unmasked input sequences as close as possible. This property is used to produce useful embeddings for masked placeholders such that usual sequence labeling architecture can be applied.

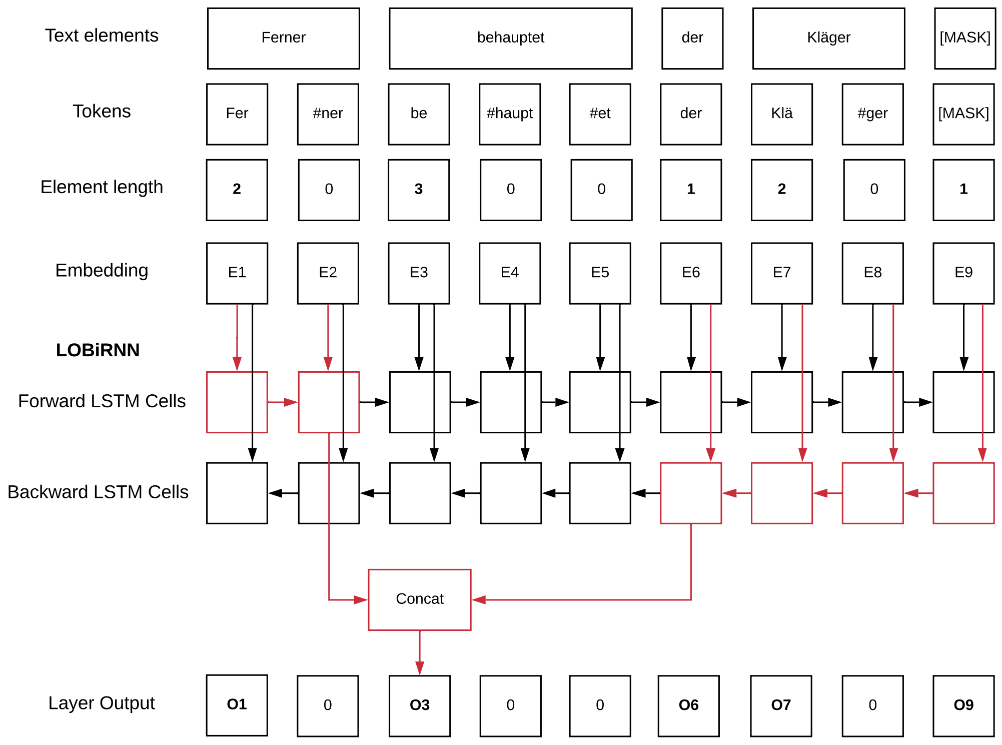

Figure 5.3.: Leave-Out bidirectional RNN Layer (LOBiRNN)

The current stat-of-the-art MLM is BERT. The contextual embeddings of the pre-trained BERT model[14] produces token vectors of dimension 768 for an input sequence with maximum length of 512. So, the preprocessing MSL has been set accordingly (see 5.3.3) and anonymization placeholders have been masked (see 5.4.2.2).

Furthermore, different feature-based [De19] training architectures have been evaluated such as convolutional and bidirectional RNN layers. On top of those additional layers, multiple dense layers have been stacked. Overall, this resulted in architectures of 2 to 5 layers on top of BERT embeddings.

For the feature-based as well as the fine-tuned models, final predictions are being calculated using a single linear output layer, followed by a sigmoid activation function.

The following table contains different variants that have been tested:

[14] https://tfhub.dev/google/bert_multi_cased_L-12_H-768_A-12/1

Variant	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5
Dense	d256	d256	-	-	-
Conv1	c15x64	c5x128	-	-	-
Conv2	c7x128	c15x64	c15x64	-	-
Conv3	c5x128	c3x64	-	-	-
Conv4	c15x16	c5x32	-	-	-
RNN1	r128	r128	-	-	-
RNN2	r256	r256	d128	d64	-
RNN3	r128	r128	r128	d128	d64
RNN4	r512	r512	d256	d128	d64
LOConv1	lc32x256	d256	d128	d64	-

Table 5.2.: Model architecture variants on top of BERT embeddings. 'd', 'c', 'r' and 'lc' denote 'dense', 'convolutional', 'biLSTM' and 'LOConv' layers. Attached numbers refer to: output feature number in case of dense layers; cell size in case of RNNs; kernel size x channel number in case of convolution.

5.4.1.3. Combined GloVe Embeddings

The GloVe word embeddings evaluated are a combination of previously pre-trained word embeddings [15], trained on German Wikipedia, and GloVe embeddings, trained directly on the corpus. This combination has been used to take advantage of general language understanding derived from a large corpus, while still profiting from lossless tokenization of SentencePiece tokens (see 5.3.3) and specialized training. The pre-trained embeddings of dimension 300 and specially trained embeddings of dimension 200 are concatenated to a final embedding of dimension 500. Outliers of the pre-trained word embedding were represented as zero vectors. In order to make use of as much contextual information as possible and reduce padding effects, a MSL of 2048 tokens has been chosen. However, because bidirectional RNNs were used, the sequences where trained independently from each other.

On top of the combined GloVe embeddings, the following leave-out layers have been evaluated:

[15] https://deepset.ai/german-word-embeddings

Variant	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5
LOConv2	d64	lc32x256	d128	d64	-
LOConv3	d128	lc32x512	d256	d128	d64
LOConv4	d64	lc64x256	d128	d64	-
LORNN1	lr256	d256	d128	-	-
LORNN2	lr512	lr512	d512	d256	-
LORNN3	lr512	lr512	lr512	d512	d256

Table 5.3.: Model architecture variants on top of combined GloVe embeddings. 'd', 'lr' and 'lc' denote 'dense', 'LOBiLSTM' and 'LOConv' layers. Attached numbers refer to: Output feature number in case of dense layers; cell size in case of RNNs; kernel size x channel number in case of convolution.

5.4.2. Regularization

Different regularization techniques have been used in order to reduce overtraining and improve generalization of the model. Furthermore, using the following techniques, the training-validation gap, further discussed in section 7.2, has been successfully reduced.

5.4.2.1. Candidate Vectors

Using BERT embeddings, placeholder tokens have been masked such that alternative vectors are predicted and the training-validation gap is reduced (see 7.2). This only works, if both placeholders, representing the positive class (anonymization), and non-placeholders, representing the negative class (no anonymization), are masked. Otherwise the model would simply recognize the correlation between mask tokens and positive labels, which is not present during evaluation. Because masking of both placeholders and non-placeholders would result in the masking of all tokens, only a subset of the sequence, referred to as candidates, are masked during training. The output for all other tokens is ignored. Those candidates always span over whole text elements, in order to ensure that there are no differences between placeholders and other tokens. Due to the limited number of positive samples, positive labels are always included in the candidate set.

While there is a mean of 0.7% of placeholder elements in the corpus, 2.0% of elements where additionally picked as candidates. This results in a total of 3.4% of masked tokens per sequence. Due to limited training data, multiple candidate vectors per sequence were created during preprocessing and randomly chosen during training.

Using combined GloVe embeddings, picking candidates is unnecessary due to the architectural properties of the leave-out layers (see 5.4.1.1).

5.4.2.2. Masking

In order to prevent the model from recognizing anonymization placeholders, which are absent during validation, candidates including placeholders are masked. Three different masking strategies were used:

- Placeholder-Only: Only the placeholder is masked using a special mask token
- Random: Candidates are masked using random token
- Mask-Token: Candidates are masked using special mask token

The BERT embeddings have been pre-trained using 10% of Placeholder-Only, 10% Random and 80% Mask-Token. In this thesis, we evaluate the following proportions:

Ref	Plhr-Only	Random	Msk-Token
M1	1.0	0.0	0.0
M2	0.1	0.9	0.0
M3	0.0	1.0	0.0
M4	0.25	0.25	0.5
M5	0.25	0.15	0.6
M6	0.0	0.2	0.8
M7	0.0	0.0	1.0

Table 5.4.: Masking variants
'Plhr-Only' and 'Msk-Token' denote 'Placeholder-Only' and 'Mask-Token'

warmup lr	10^{-2}
initial lr	10^{-5}
β_1	0.9
β_2	0.999
ϵ	1e-6

Table 5.5.: Hyperparameters of Adam optimizer

5.4.2.3. Dropout

Within all model architectures, proposed in section 5.4.1, multiple dropout layers were used directly after the embedding layers and before final dense layers. In deep neural networks, dropout layers [Sr14] randomly exclude neurones during training in order to prevent overfitting and can be applied to a large variety of architectures such as convolutional neural networks and RNNs. In this implementation, probabilities of 0.1 and 0.3 were used, depending on the number of weights.

5.4.3. Optimizer

During model training, the Adam optimizer [KB14] has been used together with a decaying learning rate. A warmup phase of 5% of total training steps [16] has been used with a learning rate of 0.01. After warmup phase, an initial learning rate is chosen and linearly decreased such that it reaches 0 in the last training step. All optimizer hyperparameters are displayed in table 5.5. Even though different optimizers and learning rates have been tried for different model architectures, this setup has delivered the overall best results.

5.4.4. Training Procedure

5.4.4.1. Hardware

The model has been trained on a single NVIDIA Tesla P100 GPU with 16280 MiB of memory. Due to the limited memory, the batch size for the fine-tuning BERT architecture had to be reduced to 6, while a batch size of 16 had been used for all other architectures.

[16]training steps refer to the number of sequences processed during training

5.4.4.2. Loss

Before being injected into the loss function, a sigmoid function was applied to the linearly combined output channels from the output layer, in order to produce values in the range between 0 and 1. As a loss function, weighted binary cross entropy has been used. Even though in practical anonymization applications high recall is preferred, the weights were chosen in such a way that neither false positives nor false negatives were preferred by the optimizer, in order to maximize accuracy, which has been used as a comparable performance metric in other anonymization tasks.[Di16]

5.4.4.3. Overfitting

Each model architecture was training using different regularization techniques until overfitting occurred. Therefore, the training data has been further split into a training set of share 90 % and a validation set of share 10 %. After every training epoch [17], the the model is validated on the validation data having been excluded from training. If the loss on the training data decreases, while the loss on the validation data increases during the next epochs, overfitting has been assumed and the training has been canceled.

5.5. Evaluation

In this section, the evaluation method is described. It is based on the test corpus, described in section 5.2. This corpus has been preprocessed by the paragraph extraction, described in section 5.3. Thereafter, the resulting paragraphs have been combined and processed using an interactive placeholder replacement program. This program can be used by a human to interactively replace anonymization placeholders with random instances of the anonymized type. Then, the documents are manually corrected so that the resulting test documents resemble non-anonymized originals as close as possible.

[17]Each epoch the whole training set is processed once

5.5.1. Interactive Placeholder Replacement

In order to replace anonymization placeholders as efficient as possible, the combined paragraphs resulting from paragraph extraction are split into text elements, which is simplified to a white-space split as described in section 5.3.2. Then, elements are processed sequentially and elements being simple words are matched using a regex and skipped. For all other elements, the user interactively decides, whether the current text element is a placeholder or not, depending on the context being printed. If it is classified as a placeholder, it is replaced with a random instance of the reference type. The type has been inferred by the user using the context of the placeholder.

The results are saved in two different files. In the first file, only decisions about text elements being placeholders is stored. This file has been used to validate the placeholder detection algorithm in section 5.3.2. The second file also contains labeled replacements and has been used to evaluate the models, after it had been corrected as described in the following section.

5.5.2. Document Corrections

After placeholders of the test corpus have been replaced using the interactive placeholder replacement program, the following mistakes have been manually changed in the document:

- Spacing mistakes, e.g. "Anw alt" to "Anwalt"
- Bracket and cite mistakes, e.g. "(see A 1 ." to "(see A 1)."
- Anonymization mistakes, e.g. names revealing the original content of placeholders

The resulting document closely resembles an original non-anonymized document and can be used to evaluate the model as described in the following section.

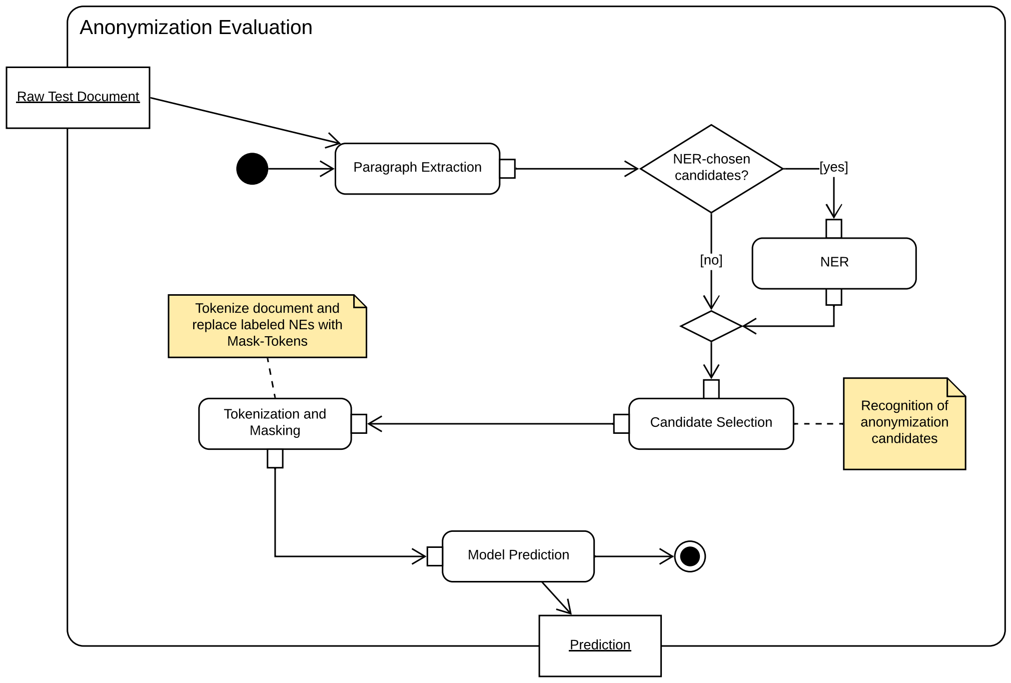

Figure 5.4.: Model Evaluation (UML 2.0 Activity)

5.5.3. Evaluation Methods

After the document has been corrected, it has been used to evaluate the models. There have been three different tasks for this purpose: Random candidates, NER-chosen candidates and evaluation without candidates. Each task is described in detail in the following sections. Before each of these test methods apply, the document is first preprocessed using the paragraph extraction, described in section 5.3, and split into into text elements. Then, depending on the method, the elements are masked and split into tokens. Finally, the model is applied and the predictions are mapped back to the original text elements. If one token within the element is classified as an anonymization, the whole element is classified as such. The resulting classifications are then compared to the true labels in order to calculate the recall, precision and accuracy metrics, which are used to evaluate the models. Figure 5.4 shows an overview of the evaluation data flow.

5.5.3.1. Using Random Candidates

Using the random candidate evaluation task, it is tested, how well the model can distinguish between anonymizations and other text elements solely depending on the surrounding context. So, randomly chosen elements and anonymizations (true positives) are masked with a special mask token and labeled as candidates, as described in section 5.4.2. This exactly equals model training using candidate vectors and masking variant M7. 2.5 % has been selected as the masking probability compared to 0.7 % of sensitive text elements in the test corpus, in order to leave enough context for prediction and also provide enough chances for false positives.

5.5.3.2. Using NER-chosen Candidates

This evaluation method equals the random candidates method, but candidates are selected using named entity recognition (see 3). So, this method evaluates, how the presented architectures are able to distinguish between sensitive and insensitive named entities (NEs). Hereby, the knowledge of a NER model having been trained on a more general context is combined with a domain-specific model, which has only been trained on the context of NEs. As an implementation, a model architecture based on LSTM RNNs [HS97] and conditional random fields (CRFs) [MH16] [HXY15] is used [18]. The model has been trained on the Germeval 2014 task [Be14]. All classified NEs are masked using a special mask token and labeled as candidates.

5.5.3.3. Without Candidates

This evaluation method has been used to test the ability of models to anonymize text without the help of other components (no NER and random pre-selection used). All elements are labeled as candidates and no token is masked. This is by far the most challenging task, since the non-anonymized input data for evaluation differs from the anonymized training data. This introduces the validation-test gap, further discussed in 7.2.

[18]https://github.com/riedlma/sequence_tagging

6. Results

In this chapter, the evaluation results of the machine learning models and the rule-based placeholder detection algorithm are presented. The evaluation methods are described in section 5.5.

6.1. Placeholder Detection Validation Results

In order to answer the first research question 1.2, a rule-based placeholder detection algorithm has been developed, described in section 5.3.2. The algorithm has been validated using the same test data set as used for the model evaluation. The results are presented in table 6.1.

Accuracy	99.9
Precision	95.9
Recall (Sensitivity)	98.0
Specificity	99.9
F1 Score	97.0

Table 6.1.: Validation results for placeholder detection
Metric values are in %

6.2. Model Evaluation Results

6.2.1. Using Random Candidates

The following table presents the evaluation results of the random candidates evaluation method. Three different models based on BERT contextual embeddings have been tested. The models have been trained using masking scheme M7, which masks candidates using a special mask token. This exactly matches the evaluation conditions. Accuracy, recall and precision metrics have been used to evaluate model performance and are solely based on the predictions of tokens labeled as candidates. The results are presented in table 6.2.1.

Embed	Variant	Cand	Mask	Epoc	Recall	Prec	Acc
BERT	FT	YES	M7	2	97.6	92.6	97.084
BERT	RNN2	YES	M7	10	93.8	88.1	94.776
BERT	RNN3	YES	M7	10	91.6	81.7	92.155
BERT	RNN4	YES	M7	10	93.0	86.6	94.051

Table 6.2.: Evaluation results for random candidates
'Embed', 'Cand', 'Epoc', 'Prec' and 'Acc' denote 'Embedding', 'Candidate', 'Training Epochs', 'Precision' and 'Accuracy'. Architecture variants are defined in tables 5.4.1.2 and 5.4.1.3. Masking schemes are defined in table 5.4.2.2. Metric values are in %

The results show that the fine-tuned (FT) BERT model outperforms the stacked LSTM-RNN models (without fine-tuning) by far on this task. This resembles the results on other NLP tasks [De19]. Also, less training epochs have been necessary to train the fine-tuned model. However, because of the model size, fine-tuning BERT requires more memory and takes about twice as long as training stacked RNNs on top of BERT (feature-based tuning). Also, the RNN2 architecture, which contains larger LSTM cells, outperforms the RNN3 architecture, which uses 3 instead of 2 biLSTM layers (see 5.4.1.2).

6.2.2. Using NER-chosen Candidates

The following table presents the evaluation results of the NER-chosen candidate evaluation method. The same models as for random candidates method have been tested, since the candidates are masked equally. The results also depend on the performance of the NER model, because only detected named entities are labeled as candidates. So, the metrics presented in table 6.2.2, match the combined performance of both models.

For this test data set, the NER model has detected only 80.05 % of replaced named entities (true positives). This limits the recall of the tested models to this percentage. Overall, 30 % of named entities detected by the NER model, need to be anonymized (positive). Therefore, 30 % represents the baseline precision.

Embed	Variant	Cand	Mask	Epoc	Recall	Prec	Acc	Det-Rec
BERT	FT	YES	M7	2	77.3	57.0	68.979	96.6
BERT	RNN2	YES	M7	10	79.1	68.9	76.941	98.8
BERT	RNN3	YES	M7	10	72.4	68.2	74.067	90.4
BERT	RNN4	YES	M7	10	74.4	68.4	75.788	92.9

Table 6.3.: Evaluation results for NER-chosen candidates
'Embed', 'Cand', 'Epoc', 'Prec', 'Acc' and 'Det-Rec' denote 'Embedding', 'Candidate', 'Training Epochs', 'Precision', 'Accuracy' and 'Detected recall'. Architecture variants are defined in tables 5.4.1.2 and 5.4.1.3. Masking schemes are defined in table 5.4.2.2. Metric values are in %

The last column of the table shows the recall values that result if only NER-detected references are considered and all undetected references (about 20%) are excluded from the metric. This is not applicable to real world tasks, since named entities have usually to be detected first. However, the values show that the models have correctly labeled almost all NER-detected references, which represents the upper limit for this approach.

As shown in the table, stacked LSTM-RNNs on top of BERT embeddings have outperformed the fine-tuned BERT model. A reason for this may be that the feature-based tuning is slightly more robust than usual fine-tuning (see chapter 7). Furthermore, the RNN2 architecture with larger LSTM cells outperforms the RNN3 architecture (see 5.4.1.2). However, a further increase in LSTM cell memory (RNN4) did not lead to an improved result.

6.2.3. Without Candidates

The following table presents the evaluation results of the evaluation without candidates. Since this by far the most challenging task, more different architectures have been trained for this task. Also, models have been trained using different masking schemes and with/without candidate labelling. Accuracy, recall and precision metrics have been used to evaluate model performance and are based on all predictions, which is presented in table 6.4.

The results show, that no model has reached both high precision and high recall. The high accuracy is justified by the low number of replaced references (true positives) compared to the high number of negatives. However, the fine-tuned BERT model using randomized masking (M3) delivered the overall best

Embed	Variant	Cand	Mask	Epoc	Recall	Prec	Acc
BERT	FT	YES	M3	4	58.1	68.9	99.352
BERT	RNN1	NO	M2	10	24.9	59.7	99.126
BERT	RNN1	YES	M3	10	88.4	15.9	95.459
BERT	RNN2	YES	M3	10	90.0	25.0	97.343
BERT	RNN3	YES	M3	15	85.4	27.6	97.735
BERT	LOConv1	NO	M3	8	27.0	54.1	99.088
BERT	Dense	YES	M2	10	18.2	63.0	99.210
BERT	Conv1	YES	M2	12	49.8	62.8	99.317
BERT	Conv1	YES	M4	35	38.2	68.1	99.317
BERT	Conv2	YES	M2	30	35.5	59.3	99.238
BERT	Conv3	YES	M4	7	25.9	72.2	99.279
BERT	Conv4	YES	M5	7	18.7	78.3	99.259
BERT	Conv4	YES	M6	5	79.7	22.8	97.507
GloVe	LOConv2	NO	M1	50	35.9	63.6	99.196
GloVe	LOConv3	NO	M1	50	41.9	64.9	99.232
GloVe	LOConv4	NO	M1	50	33.0	65.6	99.198
GloVe	LORNN1	NO	M1	50	14.3	47.3	99.034
GloVe	LORNN2	NO	M1	50	15.1	63.6	99.111
GloVe	LORNN3	NO	M1	50	14.6	66.7	99.119

Table 6.4.: Evaluation results without candidates
'Embed', 'Cand', 'Epoc', 'Prec' and 'Acc' denote 'Embedding', 'Candidate', 'Training Epochs', 'Precision' and 'Accuracy'. Architecture variants are defined in tables 5.4.1.2 and 5.4.1.3. Masking schemes are defined in table 5.4.2.2. Metric values are in %

performance. Even though, stacked LSTM-RNNs on top of BERT embeddings reached a recall of about 90%, only a very low precision of about 25% has been achieved. The convolutional architectures on top BERT performed far worse than the fine-tuned BERT model, but reached higher accuracies than the RNN approaches. The GloVe embedded leave-out convolutional architectures performed about equally well as the convolutional architectures on top of BERT embeddings. The stacked LSTM-RNN architectures on top of GloVe embeddings performed especially badly, which may suggest a lack of training data.

7. Discussion

In this chapter, the results presented in chapter 6 are further discussed.

7.1. Review of Model Evaluation Results

In general, the precision metric is less important for anonymization than the recall metric, since the neutralization of insensitive information does not cause as much harm as the revelation of a sensitive reference.

Using random candidates, the fine-tuned BERT model reached a relatively high recall of 97.6. However, the fraction of sensitive text elements (about 0.7 %) to insensitive elements (about 2.5 %) is much higher as if all text elements are classified, because, due to the limitation on test data, all positive labels are included in the evaluation. This leads to a distorted precision metric, since we have to assume that the probability of a negative text element being misclassified stays the same. Therefore, if all text elements have to be classified, both the false positive count increases, while the true positive count and the recall remain unchanged. For that reason, much higher precision values are necessary to overcome this issue. Thereby, a precision value of 92.6 % leads to an overall precision of approximately 23.9 %, which roughly matches the results of the evaluation method without candidates. This approximation is further described in the appendix A.2.

Using NER-chosen candidates, the precision metrics is more vital than the recall metrics, since the number of negative text elements has been drastically reduced and 100 % recall with 30 % precision can be reached by constant positive classification. However, because the NER model does recognize only 80 % of all sensitive named entities, the combined recall is upper-bounded by 80 %. The feature-based tuning biLSTM RNN approach (RNN2) of BERT delivers the best results with a recall of 79.1 % and a precision of 68.9 %. Thereby, it outperforms the fine-tuned BERT. Because the NER-detected named entities often includes more tokens than usually would have been anonymized, the training input is slightly different than the test input. One example for this are name titles like "Prof." and "Dr.", which are masked during evaluation, because the NER model includes those tokens in the named entity. During training, those name affixes stay unmasked, because they are usually not anonymized.

So, the results regarding the NER-chosen candidate evaluation method are promising, but a much better NER model is needed, in order to increase the recall to an acceptable value. Especially, the variety of different named entity types represents a major issue, since the test set has been made up of different kinds of references including patents and account numbers. Those named entity types are currently not supported by state-of-the-art NER models, but may be detected even using simple rule-based approaches. Nevertheless, given a more suitable NER model, the combined approach may yield applicable predictions.

Without candidates, the models have to overcome the "validation-test gap", discussed in the following section. Even though, the issue have partly been overcome, no model has managed to reach high values of both recall and precision. The reasons for that may be the data amount and quality, further discussed in the sections 7.4 and 7.5. However, the major issue may be that contextual classification cannot distinguish between named entities and entities that refer to named entities within the document. This issue is further discussed in section .

7.2. Validation-Test Gap

In the training data set, the anonymization placeholders are masked during training, because the real references behind the placeholder are unknown (this is the purpose of anonymization). In this thesis, this issue is referred to as the "validation-test gap", because the results on the validation data set (masked placeholders) during training have partly been much better than on the test set (manually replaced placeholders) during evaluation. This is only true for the unmasked evaluation method, since the other evaluation methods also make use of token masking.

As a masked language model (MLM), BERT has been pre-trained to replace vectors of masked tokens with probable vectors in respect to the context. Because BERT has been pre-trained on large non-specialized data sets, it can be fine-tuned easily on specialized data sets, drawing on its obtained general language understanding. Therefore, BERT has been used to complete the anonymized text sequences with meaningful, contextual embeddings where placeholder have been inserted. This would reduce the validation-test gap, if the embedding vectors of masked tokens are similar to the embedding of the real references.

Another approach to reduce the validation-test gap has been the use of word embeddings in combination with the introduced leave-out layers, which prevent the information flow from embedded text elements to the respective output neurons. However, even the information flow from the input embeddings to the respective output neurons is prevented, neighboring placeholders only existing in the validation data set would still influence the output. But, because the existence of nearby placeholder tokens is neither sufficient nor necessary, this fact is ignored. Instead of contextual embeddings, word embeddings have to be used, because, in general, contextual embedding vectors contain information about their neighbor tokens. However, in order to make use of general language knowledge, GloVe embeddings have been used, which have previously been trained on German Wikipedia.

In table 7.2, the test-validation gap for some of the tested architecture is displayed.

The bold rows in the table refer to test-validation gaps, while the other values in the table show that the test-validation gap has been successfully closed. Even though, the leave-out architectures on top of GloVe embeddings delivered low recall values, the results on the test set are slightly better than on the validation set, which means that this has successfully prevented the validation-test gap issue. The slight improvements may derive from the document correction of the test corpus, which reduced the noise within the documents (see 5.5.2).

Also, the random masking (M3) of tokens shows to be a sufficient way to prevent the validation test gap, if combined with candidate labeling (see RNN1,

Embed	Variant	Cand	Mask	Tst Rec	Tst Prec	Val Rec	Val Prec
BERT	FT	YES	M3	**58.1**	**68.9**	**90.4**	**94.9**
BERT	RNN1	NO	M2	**24.9**	**59.7**	**99.4**	**37.0**
BERT	RNN1	YES	M3	88.4	15.9	87.9	70.4
BERT	RNN2	YES	M3	90.0	25.0	89.3	80.3
BERT	RNN3	YES	M3	85.4	27.6	89.0	82.0
BERT	LOConv1	NO	M3	**27.0**	**54.1**	**97.7**	**25.5**
GloVe	LOConv2	NO	M1	35.9	63.6	27.4	56.2
GloVe	LOConv3	NO	M1	41.9	64.9	33.4	54.0
GloVe	LOConv4	NO	M1	33.0	65.6	28.0	56.0
GloVe	LORNN1	NO	M1	14.3	47.3	15.8	40.1
GloVe	LORNN2	NO	M1	15.1	63.6	18.0	52.0
GloVe	LORNN3	NO	M1	14.6	66.7	17.1	52.8

Table 7.1.: Comparison of metrics on validation and test data set 'Embed', 'Cand', 'Rec', 'Prec', 'Tst' and 'Val' denote 'Embedding', 'Candidate', 'Recall', 'Precision', 'Test' and 'Validation'. Architecture variants are defined in tables 5.4.1.2 and 5.4.1.3. Masking schemes are defined in table 5.4.2.2. Metric values are in %.

RNN2 and RNN3 in the table). The reduction in precision derives from the candidate labeling during validation, since the probability of positives is drastically increased. During evaluation, the chance of a negative token to be false positive stays equally high, but more negative tokens are tested, which decreases the precision, but leaves the recall unchanged.

The leave-out convolutional layer on top of BERT (LOConv1) shows very much smaller recall values for the test set compared to the validation set, even though random masking is used. The reason for this is that candidate labeling has been disabled and, in some cases, the model could infer if a randomized token has been used as the respective input, even though the leave-out layer prevents direct information flow. So, the detected random input tokens could be classified positive during training, while during evaluation no random input tokens are present.

Even though random masking without candidate labeling has been able to close the validation-test gap in the case of feature-based tuning using RNNs, the same approach has led to a large validation-test gap in the case of BERT fine-tuning. Because the BERT fine-tuning approach directly operates on the data, it may be conceivable that the model is less robust regarding changes in the input space than feature-based tuning approaches.

7.3. Downside of Contextual Analysis

The feature-based tuning of BERT using random masking (M3) and candidate labeling has resulted in high recall values, but relatively poor precision. With closer examination, the false positives often referred to text passages, where an entity is mentioned that referred to a named entity. The following example demonstrates this issue:

Example 7.3.1 *"Unstreitig hat ein Mitarbeiter der Beklagten dem Kunden «Fausner» eine Krankenversicherung angeboten. Dass die angerufene Telefonnummer für die Firma des «Kunden» in einem Branchenverzeichnis eingetragen wäre, behauptet die «Beklagte» nicht, sie mutmaßt dies nur."* [19]

In this example, the "«»"-marked text elements have been positively classified by the model. Even though only "Fausner" is a true positive, the words "Kunde" (eng.: customer) and "Beklagter" (eng.: defendant) also stand for entities that need to be anonymized. The issue is that the model cannot distinguish between named entities and entities, which have similar meaning, but do not refer to objects outside the document. Those entities could have been replaced with a named entity without changing the meaning of the text. In this case the classification would have been correct, but the context of the word would still remain unchanged. Therefore, without the information about the respective tokens itself, it is impossible to distinguish if the passage describes a names entity or truly is a named entity.

For that reason, the combination with a NER model improves the performance of those models, because passages describing named entities can be distinguished from named entities themselves.

7.4. Data Quality

In the text corpus, samples have been extracted in order to test the quality of the data, especially with respect to the consistency of anonymization, since

[19] Extract of the test document. The name "Fausner" has been inserted during interactive placeholder replacement.

this forms the basis for efficient pattern recognition. While the majority of references such as witnesses and expert witnesses (german: "Sachverständige") have been correctly neutralized, some of those obviously sensitive references could be revealed in multiple samples. Partly, this has been done through combining different named entities that have not been anonymized such as the employing organization including the job title. Partly, the named entity has been anonymized using initials and has been mentioned in some part of the court decision such that the anonymizations could be reverted. Because those court decision documents have been anonymized manually, it seems natural that those errors resulted from overlooked occurrences of named entities.

Even though the training corpus used in this work has been relatively small with respect to other NLP corpora, the corpus size has still been far too large to correct those errors manually. However, those outliers have a considerable influence on model performance.

Similar errors as described above have also been found in the test data set. During the document correction procedure, those anonymization errors in the test document have been corrected (see 5.5.2), in order to resemble the original non-anonymized legal documents as close as possible.

Finally, even though the rule-based placeholder detection algorithm has performed well during evaluation with respect to recall and accuracy, the precision of only 95.9 % adds additional noise to the data, because potential placeholders may be falsely detected as anonymization placeholders. Therefore, either more recognizable and more consistent anonymization techniques for court decisions or more accurate detection approaches are desirable to improve the quality of training data.

7.5. Data Quantity

The training corpus used in this work consists of 1.220 documents, as presented in table 5.2. In sum, those documents contain about 35.000 anonymization placeholders that are positively labeled for model training. However, due to the large variety of different reference types and different token lengths, the amount of data is considerably small. This especially impacts rare references like authorizations and bank accounts. Misclassifications of the placeholder detection algorithm and anonymization mistakes in documents (see 7.4) aggravate this issue even further.

Even though more anonymized legal documents have been available, it has not been possible to increase the size of the legal text corpus within the scope of this work, because not every legal document is suitable for training of the proposed models. The requirements on anonymized legal documents used for model training particularly include that sensitive references need to be replaced by a placeholder. Additional requirements are listed in section 5.2. In order to ensure those requirements and in particular the consistency of the anonymization method, legal documents from only one single court have been used.

8. Conclusion

Automatic legal anonymization is a highly desirable tool in order to increase the quantity of German legal data sets being publicly available. Because of the lack of available data sets containing both anonymized and non-anonymized legal documents, a machine learning approach trained solely on anonymized data is highly valuable. For this purpose, a rule-based placeholder detection algorithm has been developed and validated, in order to label anonymization placeholders in anonymized legal documents.

Furthermore, multiple different deep learning architectures have been trained using state-of-the-art generally pre-trained contextual and word-embeddings. Due to the difference between training and evaluation data, the "validation-test gap" issue has been introduced, which has caused a drop in model performance on the non-anonymized test set. This issue has been resolved using specially designed "leave-out" layers and regularization methods such as input masking and dropout layers.

The models have been evaluated on a test document corpus. This corpus has been created by manually replacing anonymization placeholders with probable named entities. Thereby, the fine-tuned BERT model approach outperformed other evaluated models in the ability to classify randomly selected text passages as sensitive or insensitive. However, purely contextual classification cannot distinguish between named entities and entities that refer to named entities within the document. So, no model reached both high recall and high precision metrics on the direct sequence classification task. Nevertheless, in combination with a generally trained NER model, the feature-based BERT tuning approach using stacked biLSTM-RNN delivered promising results, but a specialized NER model supporting more reference types is required. This shows that the sensitivity of text elements depends on both their textual context and the fact that the elements are named entities.

As a conclusion, the anonymization of German legal documents remains a complex problem and more advanced approaches are necessary in order to build fully autonomous anonymization systems. Nonetheless, contextual sensitivity classification may represent an important foundation for future anonymization systems.

9. Recommendations

In this chapter, rudiments for future work are explained, in order to use the findings of this thesis to further improve the proposed anonymization method for German legal documents.

9.1. Legal Text Corpus

As described in chapter 7, larger and higher quality data sets of anonymized German legal documents are necessary to further improve the results presented. The anonymized text corpus should have the following properties to be optimally suitable for contextual sensitivity classification:

1. Anonymization should follow consistent and legally defined rules
2. All anonymized references should be replaced by placeholders without further modifications
3. Anonymization placeholders should be consistent and distinguishable from other text fractions, e.g. using "«»"-Notation
4. Text should be divided into enumerated paragraphs of limited length
5. Meta-data such as case ID and court name should be easily extractable and separable from the rest of the document
6. Spacing and spelling mistakes should be minimized

9.2. Specialized NER

The development of a legal NER model would further enhance the proposed anonymization method, which combines contextual sensitivity detection with NER. The developed NER model should support a more complete and specialized set of named entity types than current unspecialized NER solutions. The following list contains all named entity types that have been present in the test corpus, but are currently not supported by most pre-trained NER models:

1. Patents, e.g. "DE102006011652A1"
2. Brand numbers, e.g. "30 2017 029 616"
3. Brand names, e.g. brand "Coca Cola"
4. Authorizations, e.g. building license "602-1.2-2012-3304-23"
5. Websites, e.g. "www.example.org"
6. Email addresses, e.g. "user@example.org"
7. Account number, e.g. "DE91 1000 0000 0123 4567 89"
8. Contract numbers, e.g. loan agreement number "738263528"
9. Media names (Magazines, Newspapers, TV Shows, Newsletters), e.g. magazine "GEOlino"

Furthermore, a specialized NER should only detect sensitive named entity parts and exclude name affixes such as "Prof." and "Dr." as well as legal forms such as "GmbH" and "AG".

Using the proposed methods of this work, the recognition of a named entity is a necessary condition in order to classify it as sensitive. Therefore, specialized NER should be optimized with respect to high recall, since false positive predictions may be compensated by the contextual sensitivity classification.

9.3. Text Splitting Improvements

In this work, text elements have been defined as the smallest textual unit that is be able to contain a full reference to an object outside the document. Those elements have been used as the textual classification unit for sensitivity. For practical reasons, the process of splitting text into text elements has been simplified to the common white-space split. However, some reference types span over multiple whitespaces in such a way that individual text elements do not contain full references. Because text elements represent the basis for masking, this introduces issues during training, which have been ignored in this work. Furthermore, if single references consist of multiple text elements, all elements of a reference have to be classified positively in order to anonymize it correctly. This further complicates the anonymization process.

As future work, a rule set for text element splits may be developed such that each anonymizable reference consists of exactly one element. The following list contains a selection of special instances, that are especially challenging. Text elements containing references are labeled using "«»"-Notation:

- Names including special symbols:
  E.g. "«Fentler + Sohn GmbH»", "Anwalt «Thilo von Schmiedeberg»"
- Numbers being split for readability:
  E.g. "Bankkonto «DE91 1000 0000 0123 4567 89»"
- Dates: E.g. "«25. September 1996»"

9.4. Language Models Improvements

The state-of-the-art masked language model (masked LM) BERT [De19] has been used in this work to infer vector representations from placeholders. However, the masked LM has not produced contextual word vectors for masked text elements that were indistinguishable from unmasked representations. One reason for this is that BERT has been trained by masking only single tokens instead of whole "words" or elements. Even though this does not represent a formal requirement for masked LMs, this has led to the discussed validation-test gap using special mask tokens (see 7.2). Furthermore, the existence of mask tokens has changed the embedding vectors such that prediction accuracy has been diminished for non-anonymized input sequences using naive model approaches (without random masking).

This suggests the development of an improved masked LM, which produces embeddings that meet the following special requirements:

1. Word embeddings depend only on the surrounding context of tokens
2. Embeddings encode most important syntactic and semantic information
3. Produced embedding vectors of masked text elements are indistinguishable from embeddings of unmasked text elements

Bibliography

[Be14] Benikova, D.; Biemann, C.; Kisselew, M.; Pado, S.: *GermEval 2014 Named Entity Recognition Shared Task: Companion Paper*. March 2014. 3, 5.5.3.2

[Da19] Dai, Z.; Zhilin, Y.; Yang, Y.; Carbonell, J.; Le, Q.; Salakhutdinov, R.: *Transformer-XL: Attentive Language Models Beyond a Fixed-Length Contex*. June 2019. 5.3.1

[De19] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K.: *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 2019. 2.2.1, 2.2.2.1, 5.3.3, 5.4.1.2, 6.2.1, 9.4

[Di16] Dias, F.: *Multilingual Automated Text Anonymization*. June 2016. 3, 3, 5.4.4.2

[DMB16] Dias, F.; Mamede, N.; Baptista, J.: *Automated Anonymization of Text Documents*. July 2016. 3

[E1] Eßer, D.: *Die Veröffentlichungspflicht der Gerichte und das Internet. JurPC*. 119. 2001. 1.1

[Fr17] Frauchiger, D.: *Anwendungen von Design Science Research in der Praxis*. June 2017. 4

[HS97] Hochreiter, S.; Schmidhuber, J.: *Long Short-term Memory*. December 1997. 3, 5.4.1.1, 5.4.1.1, 5.5.3.2

[HXY15] Huang, Z.; Xu, W.; Yu, K.: *Bidirectional LSTM-CRF Models for Sequence Tagging*. August 2015. 5.5.3.2

[KB14] Kingma, D.; Ba, J.: *Adam: A Method for Stochastic Optimization*. December 2014. 5.4.3

[Ku18] Kudo, T.: *Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates*. April 2018. 2.2.1

[La16] Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C.: *Neural Architectures for Named Entity Recognition*. March 2016. 3

[Me06] Medlock, B.: *An Introduction to NLP-based Textual Anonymisation*. January 2006. 2.1, 3

[MH16] Ma, X.; Hovy, E.: *End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF*. May 2016. 5.5.3.2

[Op17] Opijnen, M. v.; Peruginelli, G.; Kefali, E.; Palmirani, M.: *On-line Publications of Court Decisions in the EU*. February 2017. 1.1

[PSM14] Pennington, J.; Socher, R.; Manning, C.: *GloVe: Global Vectors for Word Representation*. October 2014. 2.2.2.2, 3, 5.3.3

[SHB16] Sennrich, R.; Haddow, B.; Birch, A.: *Neural Machine Translation of Rare Words with Subword Units*. August 2016. 2.2.1

[SKU14] Stubbs, A.; Kotfila, C.; Uzuner, : *Automated systems for the de-identification of longitudinal clinical narratives*. June 2014. 3

[Sr14] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.: *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. June 2014. 5.4.2.3

[Sw07] Sweeney, L.: *Replacing Personally-Identifying Information in Medical Records, the Scrub System*. September 2007. 3, 3

[Tv04] Tveita, A.; Edsberga, O.; Røsta, T.; Faxvaaga, A.; Nytrøa, ; Nordgardd, T.; Ranangc, M. et al.: *Anonymization of General Practioner Medical Records*. January 2004. 3, 3

[ULS07] Uzuner, ; Luo, Y.; Szolovits, P.: *Evaluating the State-of-the-Art in Automatic De-identification*. September 2007. 3

[Va17] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. et al.: *Attention Is All You Need.* June 2017. 2.2.2.1

A. Appendix

A.1. Anonymization Placeholder List

The placeholders of the legal document corpus have been divided into obvious placeholders and potential placeholders (see 5.3.2). While obvious placeholders can have one meaning only, potential placeholders do not unconditionally need to represent anonymized references.

The following list contains all obvious placeholder patterns used:

- Special ellipses: '...-suffix', '. . . ', 'a. . . '
- Same character: 'Xxxxx', 'Yyy', 'Zzzz-suffix'
- Letters in quotes: '"A."', '"a"'
- Elipsis in quotes: '"..."', '". . . "'
- Single letter url: 'http://a', 'www.x.de'
- Single letter email: 'a@x.de', 'user@l.de'

The following list contains all potential placeholder patterns used:

- Simple ellipses: '...', '"..."'
- Single letter: 'a', 'A', 'A.', 'A-suffix'
- Same characters in quotes: '"Aaaa"', '"XXXX"'

A.2. Random Candidate Precision Approximation

The true precision of evaluation using random candidates without masking can be approximated as follows:

Let $X, T, M \in \{0, 1\}$ be random variables.
Let $X = 1$ denote the event that the text element is really positive.
Let $T = 1$ denote the event that the text element is predicted positive.
Let $M = 1$ denote the event that the text element has been masked.

We are interested in $P(X = 1|M = 1)$ being the true precision. It holds:

$$\begin{aligned} P(X = 1|M = 1) = P(X = 1|T = 1, M = 1) \cdot P(M = 1|T = 1) \\ +P(X = 1|T = 1, M = 0) \cdot P(M = 0|T = 1) \end{aligned} \tag{A.1}$$

The term $P(X = 1|T = 1, M = 1)$ is the presented evaluation precision. Because all true positives are included in the candidates,
$P(X = 1|T = 1, M = 0) = 0$. It follows:

$$P(X = 1|M = 1) = P(X = 1|T = 1, M = 1) \cdot P(M = 1|T = 1) \tag{A.2}$$

This means that we need to calculate the factor $P(M = 1|T = 1)$:

$$P(M = 1|T = 1) = \frac{P(T = 1|M = 1) \cdot P(M = 1)}{P(T = 1)} \tag{A.3}$$

We know the masking probability $P(M = 1)$ and
$P(T = 1|M = 1) = \frac{TP+FP}{TP+TN+FP+FN}$.

Now, we assume $P(T = 1|X = 0, M = 0) = P(T = 1|X = 0, M = 1)$, since the masked negative elements have been randomly chosen:

$$\begin{aligned}
P(T=1) = P(T=1|X=1, M=1) \cdot P(X=1|M=1) \cdot P(M=1) \\
+P(T=1|X=0, M=1) \cdot (P(X=0|M=1) \cdot P(M=1) \\
+P(X=0|M=0) \cdot P(M=0)) \\
= P(T=1|X=1, M=1) \cdot P(X=1|M=1) \cdot P(M=1) \\
+P(T=1|X=0, M=1) \cdot P(X=0)
\end{aligned} \tag{A.4}$$

Finally, all values in equation A.4 can be directly calculated using the evaluation data.