




Anonymization of German financial documents using neural network-based language models with contextual word representations

David Biesner^{1,2}  · Rajkumar Ramamurthy¹ · Robin Stenzel¹ · Max Lübbering¹ · Lars Hillebrand^{1,2} · Anna Ladi¹ · Maren Pielka¹ · Rüdiger Loitz³ · Christian Bauchhage¹ · Rafet Sifa¹

Received: 1 October 2020 / Accepted: 4 September 2021

© The Author(s) 2021

Abstract

The automatization and digitalization of business processes have led to an increase in the need for efficient information extraction from business documents. However, financial and legal documents are often not utilized effectively by text processing or machine learning systems, partly due to the presence of sensitive information in these documents, which restrict their usage beyond authorized parties and purposes. To overcome this limitation, we develop an anonymization method for German financial and legal documents using state-of-the-art natural language processing methods based on recurrent neural nets and transformer architectures. We present a web-based application to anonymize financial documents and a large-scale evaluation of different deep learning techniques.

Keywords Neural nets · Transformers · BERT · RNN · Financial documents · Anonymization · Sequence tagging

1 Introduction

The automatic processing of text documents has become of vital importance in several industrial applications. The availability of digital financial and legal documents is increasing and companies rely on automated methods for handling and analysis, often based on or assisted by machine learning tools. The development of such tools usually requires researchers and developers to have access to documents as part of data exploration or the model training pipeline. However, such financial data typically cannot be processed or shared beyond authorized parties due to the prevalence of sensitive information regarding specific individuals and organizations, which significantly restricts development even within the organization. One possible solution is to perform

either pseudo-anonymization or full anonymization of data before further processing.

After removing names, locations, dates and other entities that make the inference of personal information possible, one remains with a document that is safe to distribute but still contains the original structure and language, leaving it suitable for analysis, training and prediction.

Even when anonymization of data is a direct interest to a business or even a legal necessity (see Sect. 1.1 for examples), manual anonymization is often unfeasible due to the sheer amount of classified documents and the necessity for the human anonymizers to have authorized access to the original documents. In this work, we propose a deep learning-based framework for automatic anonymization of text documents and study its effectiveness on financial documents.

1.1 Legal context

While the principle of anonymization is simple, concrete applications must follow narrow legal guidelines which we want to elaborate on for the European and German market.

With the introduction of the General Data Protection Regulation, (GDPR)¹ personal data can only be further processed

David Biesner and Rajkumar Ramamurthy have contributed equally to this work.

✉ David Biesner
david.biesner@iais.fraunhofer.de

¹ Fraunhofer IAIS, Sankt Augustin, Germany

² University of Bonn, Bonn, Germany

³ PriceWaterhouseCoopers GmbH WPG, Frankfurt, Germany

¹ <https://gdpr-info.eu/>.

if they are compatible with the very strict purposes permitted by law for which this data were collected.² These purposes usually do not include the usage of the collected data for the training of machine learning tools. In fact, the GDPR does not even mention the processing of “Big Data” or algorithms with a single-word [1]. This does not change with the 2019’s entry into force of a new regulation of the EU on the free flow of non-personal data. As the name already suggest, this regulation allows the storage and processing of data across the Member States without unjustified restrictions, as long as the data are not personal. However, the principle of purpose limitation is not applicable once the data are anonymized,³ and therefore, this data can be used for developing digital solutions across Europe.

Furthermore, if the personal data are no longer necessary for the purpose for which it was collected, the GDPR grants the data subject a “right to be forgotten,” i.e., the right that its data are being erased.⁴ In practice, a company that collects personal data, like every service provider, would need to delete their customer contracts at the time of its termination date. However, this could contradict legal retention periods, for example, for tax purposes. This may be avoided, if the company anonymizes their contracts at the termination date. Considering the amount of the corresponding documents, manual anonymization is not appropriate under these circumstances.

However, the demand for anonymization of confidential data has always been present, not only since the introduction of the GDPR. For instance, publication of judgments in the public interest is, at least in Germany, a direct constitutional task for the judicial power and therefore for every single court.⁵ However, these publications need to be anonymized, regardless of the GDPR, to protect the fundamental right to informational self-determination.⁶ Until now, such anonymization is mainly done manually, resulting in a publication of only a mere fraction of the judgments that are in the public interest.

1.2 Our contributions

All of the examples above have in common that the data with the need for anonymization is usually part of documents like contracts or other reports. Consequently, we address this concern of data privacy and protection and present a web-based anonymization application that anonymizes sensitive information such as names of persons, locations, organizations, numbers, telephone numbers, dates and URLs in a piece of

² Art. 17 GDPR.

³ Recital 26 GDPR.

⁴ Art. 5 GDPR.

⁵ BVerwG, 26.2.1997 – 6 C 3/96.

⁶ Art. 2 Abs. 1 GG in conjunction with Art. 1 Abs. 1 GG.

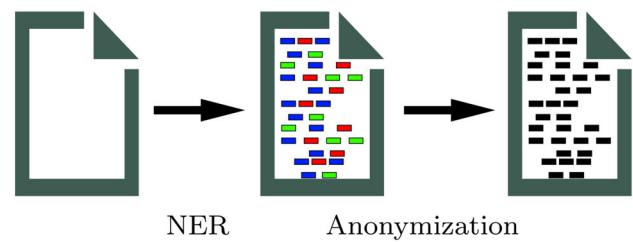


Fig. 1 General workflow for anonymizing a document using named entity recognition. First, sensitive entities are identified using deep learning methods and rule-based post-processing. Then, the identified entities are replaced with appropriate tags to preserve the text structure or hidden behind a general anonymized tag

writing by the example of financial documents. We tackle this using state of the art deep learning and natural language processing techniques as well as rule-based post-processing. A general outline of the workflow is shown in Fig. 1.

Our main contributions in this work are:

- A method to anonymize 99% of all sensitive entities contained in German financial documents while maintaining high readability and preserving the structure of the given text
- Presenting a web-based application and an API to use our method on various types of documents
- A quantitative evaluation of multiple state-of-the-art deep learning techniques for anonymization as well as the impact of domain-specific language models for financial documents.

Note that a preliminary version of this work was presented (unpublished) at an AAAI-20⁷ workshop. This version of the paper includes discussion of a new type of deep-learning architecture (see Sect. 4.1.3) with theory, details on training and new experimental results.

2 Related work

Earlier systems on anonymization focused primarily on medical records. The first anonymization system was developed by [2] used several pattern matching algorithms which detect names, phone numbers, etc. Later in 2006, a challenge was hosted to anonymize clinical data which were also made available as public dataset, namely i2b2⁸ for de-identification. Several systems were developed as a result of this challenge which tackled the problem using named-entity recognition [3,4], rule-based systems [5] and hybrid system [6] which uses look-up on dictionaries, regular expressions

⁷ <https://aaai-kdf2020.github.io/>.

⁸ <https://www.i2b2.org/>.

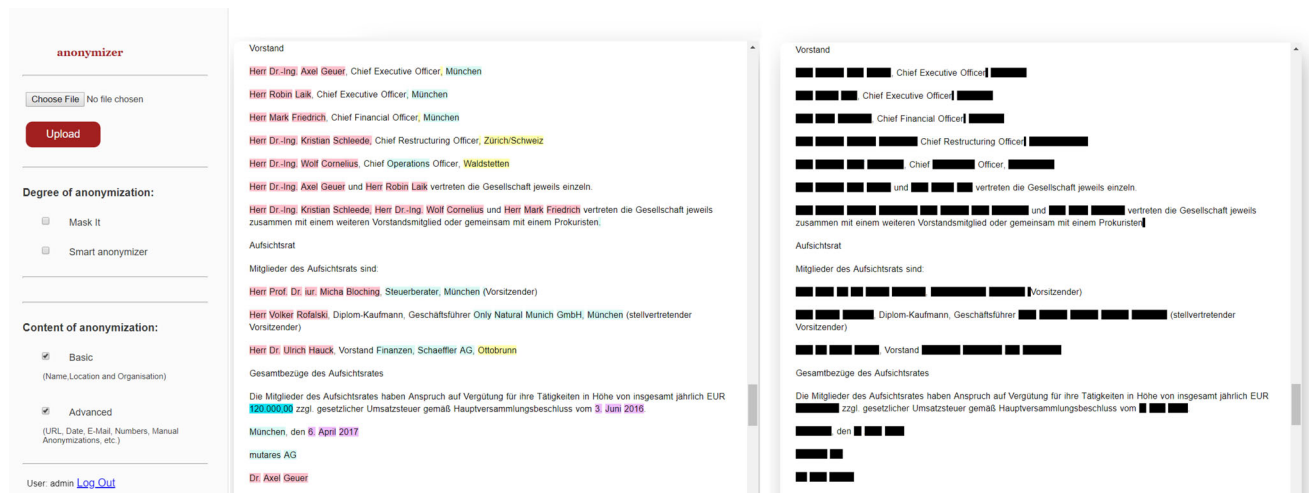


Fig. 2 A screenshot of our anonymization tool; the left pane contains the UI controls for uploading the document and other settings such as to turn on the anonymization for numbers and to enable masking. To the right of it, there is the document pane and it shows the content of the

document in which sensitive entities are highlighted if the mask option is not selected. If the mask option is selected, then the document pane shows the same content instead with sensitive entities masked

and as well as model-based classifiers. To the best of our knowledge, we present the first large scale of evaluation of anonymization techniques with respect to financial documents.⁹

3 Application

3.1 Web-based application

A screenshot of the application is shown in Fig. 2. It is a web-based application (implemented via the *Flask*¹⁰ framework) which allows the user to upload text documents (e.g., *.docx*, *.pdf*, *.txt*, *.json*) and visualize the anonymized content. The interface contains two panes; a left pane with controls and a right pane where the anonymized document is rendered. There are two basic configurable settings: by default, names, locations, organizations and other entities are anonymized using our deep learning methods. Additionally, one can enable anonymization of numbers, dates, etc., which are detected using regular expressions. The sensitive entities are highlighted in different colors based on their types; In Fig. 2, the names of persons, companies and locations are highlighted in red, green and blue, respectively. Further, the tool allows the user to enable masking such that sensitive entities are blacked out entirely as shown in the figure on the rightmost pane.

Parsing the original document allows for replacement of text within the document format (e.g., *.docx* implemented using the *python-docx*¹¹ python library, *.xlsx* using the *openpyxl*¹² library) while keeping formatting like text size, fonts and layout intact. Once a document is processed, the tool lets the user download an anonymized version of the document in the original format (e.g., *.docx*), in which all relevant entities are replaced by generic tokens (e.g., <PER>, <ORG>, <LOC>, ...).

Additionally, the tool anonymizes *.pdf*-documents and application of OCR methods (*pytesseract*¹³ library) allow for anonymization of scanned *.pdf* files.

All machine-learning related work was implemented using the *pytorch*¹⁴ framework.

3.2 API

Since the main application of this tool is document pre-processing for further distribution or use in the training of machine learning systems, we desire the anonymization of an entire document corpus. These anonymized documents can afterward be handled by developers without clearance for the original data. For this reason, we provide a REST API and python package for internal usage. This makes it possible for an employee with the required clearance for the original documents and no involvement in the development

⁹ “Prof. Dr. Vogel ...Mr. Vogel ...vogel@company.com ...Munich, April 6th 2017.”

¹⁰ <https://pypi.org/project/Flask/>.

¹¹ <https://pypi.org/project/python-docx/>.

¹² <https://pypi.org/project/openpyxl/>.

¹³ <https://pypi.org/project/pytesseract/>.

¹⁴ <https://pytorch.org/>.

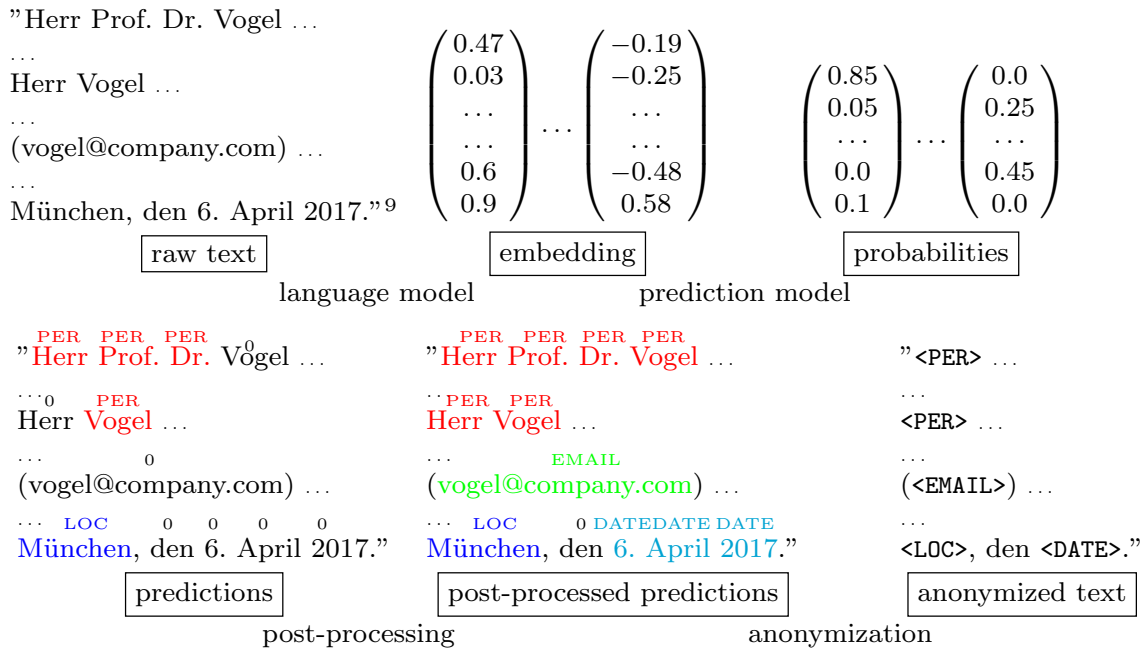


Fig. 3 Workflow from raw text to final anonymized output. We convert each token into a numerical vector using a trained language model, use a neural net classifier to predict probabilities for each class for each token,

choose the class with the highest probability, apply post-processing and finally replace named entities with corresponding labels in text, leaving words classified as 0 intact

process to use the tool to anonymize a corpus of documents at once and return the anonymized data. This leaves a readable text without sensitive information that can be further analyzed by different machine learning approaches.

4 Anonymization as sequence tagging

We tackle the problem of anonymization as a sequence tagging task [7]. Given a document consisting of several sentences in which each sentence is a sequence of words (tokens), our goal is to assign a suitable label to each token indicating if it contains sensitive information or not.

The possible labels include

- 0 (contains non-sensitive information),
- *ORG* (contains an organization or part of an organizations name),
- *PER* (contains a person or part of a persons name),
- *LOC* (contains a location or part of a location name),
- *PROD* (contains a product name),
- *SEG* (contains information about the industry of the company),
- *URL* (contains an URL),
- *TEL* (contains a phone number),
- *DATE* (contains a date),
- *NUM* (contains a number),

- *EMAIL* (contains an e-mail address) and
- *OTH* (contains any other sensitive information).

In particular, we refer to *ORG*, *PER*, *LOC*, *PROD*, *SEG* and *OTH* as *named entities* as it is part of the well-known problem of named entity recognition [8] in natural language processing.

We employ a multi-step approach as depicted in Fig. 3. **Step 1:** Predict the named entities in each document using language models and deep learning methods. **Step 2:** Make these predictions consistent across each document. **Step 3:** Predict the remaining labels using rule-based classifiers and assign to the respective tokens. **Step 4:** Replace the text of tokens by appropriate tags in order to preserve the sentence structure and semantics.

4.1 Language models

4.1.1 Word embeddings and contextual language models

Unlike traditional string-based methods (e.g., rule-based systems using *regex*), modern deep learning approaches for text classification require a two-step approach; first, the raw text has to be converted into a numeric representation, usually a vector of fixed dimension for each word in the text. The numeric representation of a token is then fed into a classifier that outputs probabilities for each class.

Word representations can be obtained in two forms: **global word embeddings** and **contextual word embeddings**. Global word embeddings provide numeric vectors for each word in a vast vocabulary and they are obtained using a large corpus of language data to capture semantic information of each word. Typically, these embeddings are trained to satisfy some distance metric (e.g., Euclidean distance or cosine similarity) between semantically similar vectors. For example, the word vector corresponding to *finance* would be closer to the vector for *banking* than to the vector corresponding to *apple*. Popular word embedding models include word2vec [9] and glove [10]. The advantage of these models lies in their ease of use that they can be distributed as text documents containing words and corresponding vector weights. And retrieving an embedding for a certain word simply requires just a lookup of the corresponding entry in the list of vectors. However, the reliance on exactly one vector per word has a major disadvantage that the same word can have multiple meanings depending on context, which cannot be captured by these global embeddings.

Consider the following two sentences

- “Herr Vogel ist Geschäftsführer der Test GmbH.”¹⁵
- “Der frühe Vogel fängt den Wurm.”¹⁶

The word *Vogel* refers to a bird in one sentence and a person in the other. A global word embedding model would retrieve the same vector for both tokens and an anonymization model based on individual word embeddings would either anonymize the animal or let the name pass through. A prediction model that takes as input a sequence of word embeddings, as they appear in the sentence, might be able to differentiate the meanings in this context. However in this work, we only consider prediction models based on single-word embeddings.

In contrast, contextualized language models offer embeddings that include context for each word. Like word embeddings, these models are also pretrained on a large corpus of language data, but are based off neural networks themselves that process each sentence to capture the context. In the example mentioned above, the language model would capture the context of the sentence (see sections below for details on how) and calculate distinct word embeddings for both instances of the word *Vogel*. An anonymization model could then learn from these contextual word embeddings to anonymize the appropriate name. However, this means that the retrieval of word embedding which is part of the prediction pipeline is not a simple dictionary lookup, but rather a deep learning model itself that can vastly exceed the prediction model in size and complexity. In our experiments, the

retrieval of the contextualized word embeddings takes up the majority of the processing power and inference time.

In our setting, the most important distinction between classic word embeddings and language models is the handling of out-of-vocabulary words. Though global word embeddings offer vectors for large vocabularies of words (e.g., *Glove* with up to 400,000 words), there is no guarantee that for names of persons, locations and companies there even exist an embedding. While there are several ways, depending on the task, of dealing with these missing embeddings, obtaining a reasonable embedding for each token naturally is definitely preferable.

In contrast, contextualized language models work with a vocabulary of either characters or so-called sub-word tokens [11]. In either case, the vocabulary contains each character that is needed to construct words in a given sentence. Therefore, a contextualized language model is able to embed any word, no matter how common or rare.

These theoretical considerations hold up in practice, where architectures based on contextual language models severely outperform traditional approaches based on word embeddings. For instance, [12] report a F1-score of 76-79% on the CONLL-2003 task [13], compared to 91% reported in [14] on the same task. In a similar work on German NER [15], the use of contextual embeddings [14] obtained better performance when compared to using only the Fasttext word embeddings [16]. For these reasons, we do not consider classical word embeddings for our task and refrain from an additional evaluation of these methods on our dataset.

4.1.2 Recurrent neural net-based language models

In our work, we utilize *flair* [14], which employs a bi-directional character-based recurrent neural net that traverses each sentence in both forward and backward direction, which is trained to predict the next character conditioned on the ones it saw before. In order to predict the beginning of the next word or the next character in a word, it needs information on the sentence context that will be stored in the hidden states of the network layers. The corresponding hidden states of the network at the beginning and end of each token together act as the numeric vector representation for that token. It contains both information of the word itself and an encoding of the surrounding words, thereby capturing the context of the token.

In this paper, we evaluate several versions of this language model that differ both in training data (i.e., what language corpus the model was trained on) and their size, referring to the dimension of the output vector. A smaller language model outputs a smaller token vector that stores less information but can process a document significantly faster. See Fig. 4 and Table 1 for a quantitative evaluation of language models of different sizes.

¹⁵ “Mr. Vogel (bird) is CEO of Test GmbH (equiv. LLC).”

¹⁶ “The early bird catches the worm.”

4.1.3 Transformer-based language models

In recent years, the development of the transformer model [17] has led to many breakthroughs in natural language processing. Transformer-based architectures rely almost entirely on self-attention, which processes the sequence of words as a whole and considers relationships between all pairs of tokens in the sentence. This architecture allows for tracking long dependencies in text, which may be an issue with recurrent neural net-based architectures that lose their “memory” of processed words rather quickly [18].

One very popular transformer-based architecture is *BERT* [19]. *BERT* trains a transformer-based neural net model by masking random tokens in a sentence and trying to reconstruct them. While recurrent architectures, as described above, receive the entire sentence to one side to reconstruct the next token, *BERT* receives the entire sentence context except the tokens that needs to be reconstructed. Additionally, the same model architecture can be trained on many tasks like language modeling (i.e., token reconstruction), translation and token or sequence classification. This way researchers are able to train a single model on various datasets to improve general language understanding within the model.

This type of architecture has led to new state of the art results, for instance in machine translation. However, one drawback of *BERT* is a reliance on a maximum sequence length of 512, which other models are able to overcome [20].

4.2 Prediction models

After obtaining the token representations using the language models, the text is fed into the classifier network as an ordered list of numeric vectors, one for each token, which is then subsequently mapped onto corresponding probabilities for each of the 7 named entities (*O*, *ORG*, *PER*, *LOC*, *PROD*, *SEG* and *OTH*). During training, the network is trained to predict the expert annotated labels for each token by minimizing the cross-entropy loss. Once the network is trained in this fashion, during inference, the label with the highest probability is predicted. We consider three different classifiers architectures:

4.2.1 MLP

First, we consider a simple fully connected network (*multi-layer perceptron*) that takes each token representation individually, passes it through several layers and outputs probabilities for each of the 7 named entities. In this case, the prediction for each token is treated independently and relies solely on the contextual representation provided by the language model. This classifier is preferred because of faster inference time and easier interpretability of results.

4.2.2 RNN

Although a simple MLP is sufficient to classify a token since the representation contains the context, it is still beneficial to process the text using a *recurrent neural net* which further enhances the context and more importantly the required span of context can be trained for the given task. For this reason, we consider a bi-directional variant of *Long Short Term Memory (LSTM)* [21] which traverses the list of vectors in both directions, processing stored context information from previous tokens and the current token. The outputs along both directions (forward and backward) are concatenated and passed through a final fully connected prediction layer mapping to probabilities for each of the 7 named entities.

4.2.3 RNN + CRF

With MLP and RNN, the prediction of each token is treated independently. In order to incorporate dependencies between predicted labels, the fully connected layer from the output states of the RNN to the output layer can be replaced by a *conditional random field (CRF)* [22] that learns a mapping of sequences of representations taking into account the predicted labels of consecutive tokens.

4.3 Post-processing

As discussed in Sect. 3.1, we also provide an option in our application to anonymize URLs, dates, numbers and e-mail addresses. Since they mostly have regular patterns, we have implemented regular expressions to detect these entities.

For the task of anonymization, we want to give higher preference to recall than precision, since anonymizing too many words is preferable to missing a word that should be anonymized. Due to the context dependence of the applied language and prediction models, there might be tokens in the given text which are predicted as sensitive in one place and as not sensitive in other places. To this end, we propose the application of a post-processing step that ensures consistency in the predicted labels: a token (e.g., a persons name) that is predicted as a named entity once in the document is always replaced by the corresponding label, even if the classifier predicted it as non-sensitive in another sentence.

5 Experiments and results

5.1 Datasets and models

In the following subsections, we describe the specific datasets, architectures and techniques used for training language models and classifiers.

Table 1 Quantitative evaluation of all described language models and classifiers on the NER evaluation dataset of financial documents and the GermEval dataset

Architecture	Embedding	On financial documents						On Germeval		
		Before post-processing			After post-processing			Before post-processing		
		Precision	Recall	F ₁	Precision	Recall	F ₁	Precision	Recall	F ₁
MLP	BANZ1024	0.989	0.485	0.651	0.960	0.682	0.797	0.928	0.076	0.140
MLP	BANZ2048	0.986	0.584	0.734	0.954	0.777	0.856	0.938	0.136	0.238
MLP	BANZ4096	0.986	0.765	0.862	0.938	0.867	0.901	0.923	0.179	0.300
MLP	flairDE	0.967	0.933	0.950	0.905	0.968	0.935	0.669	0.793	0.726
MLP	BERT	0.975	0.617	0.756	0.923	0.793	0.853	0.778	0.193	0.309
RNN	BANZ1024	0.958	0.948	0.953	0.897	0.976	0.935	0.720	0.646	0.681
RNN	BANZ2048	0.963	0.966	0.964	0.907	0.985	0.944	0.778	0.622	0.691
RNN	BANZ4096	0.972	0.969	0.970	0.915	0.986	0.949	0.815	0.638	0.716
RNN	flairDE	0.966	0.968	0.967	0.906	0.988	0.945	0.808	0.848	0.828
RNN	BERT	0.958	0.957	0.957	0.887	0.978	0.930	0.775	0.573	0.659
RNN_CRF	BANZ1024	0.960	0.961	0.960	0.897	0.983	0.938	0.741	0.684	0.711
RNN_CRF	BANZ2048	0.968	0.969	0.968	0.910	0.987	0.947	0.784	0.654	0.713
RNN_CRF	BANZ4096	0.971	0.973	0.972	0.910	0.987	0.947	0.796	0.675	0.731
RNN_CRF	flairDE	0.968	0.970	0.969	0.903	0.990	0.945	0.824	0.840	0.832
RNN_CRF	BERT	0.955	0.950	0.952	0.878	0.975	0.923	0.743	0.582	0.653
flairNER	flairDE	0.938	0.779	0.851	0.853	0.897	0.874	0.889	0.755	0.817

We provide all metrics on the positive class (*PER*, *ORG* and *LOC*). The best performance for each metric is marked bold for each column, respectively. Post-processing for these classes only consists of ensuring label consistency. We do not evaluate post-processing for Germeval since its structure (independent sentences) does not fit our post-processing methods

5.1.1 Language model corpus

As discussed in the previous section, in order to obtain contextual representations for tokens, we consider different language models. The baseline model that we use is a pre-trained language model provided by the flair framework which is trained on a large general corpus of German sentences consisting of 500 million words. We refer the embedding obtained using this model as *flairDE*. The language corpus used in the training of this embedding might cause licensing issues, e.g., the Wikipedia corpus is distributed under *GNU Free Documentation License* and *Creative Commons Attribution-Share-Alike 3.0 License*, which prohibit commercial use without adapting the same license to the project. Additionally, a language model trained on data that is similar to the financial text might provide an advantage over a language model trained on general language data and a custom language model allows for tuning the embedding size in order to optimize runtime. We therefore train language models on a corpus of language data from *Bundesanzeiger*¹⁷ (*BANZ*), consisting of 19,000 German financial documents (200 million words).

¹⁷ <https://www.bundesanzeiger.de/ebanzwww/wexsservlet>.

5.1.2 Document corpus

We train our deep learning classifier models using a corpus of 407 published German financial documents, annotated manually by domain experts. We split the dataset into 305 training and 102 validation documents. Once a model is trained, we provide a final evaluation dataset consisting of 45 thoroughly annotated documents. This evaluation dataset contains a total of 189k tokens, 17k (9.1%) of which belong to one of the classes *ORG*, *LOC* and *PER*. In order to provide results comparable to other NER and anonymization projects, we additionally evaluate all trained models on the *GermEval 2014 NER Shared Task* corpus [23], consisting of 29k sentences annotated for NER with a total of approximately 590k tokens, 8.4% of which are named entities.

5.1.3 RNN-based language models

To train and use a language model on our data, we employ the framework provided by the *flair* python-package.¹⁸ It implements a bidirectional LSTM on a character level. We train language models on the *BANZ*-corpus with 1024, 2048 and 4096 dimensions. These are denoted by *BANZ1024*,

¹⁸ <https://github.com/zalandoresearch/flair>.

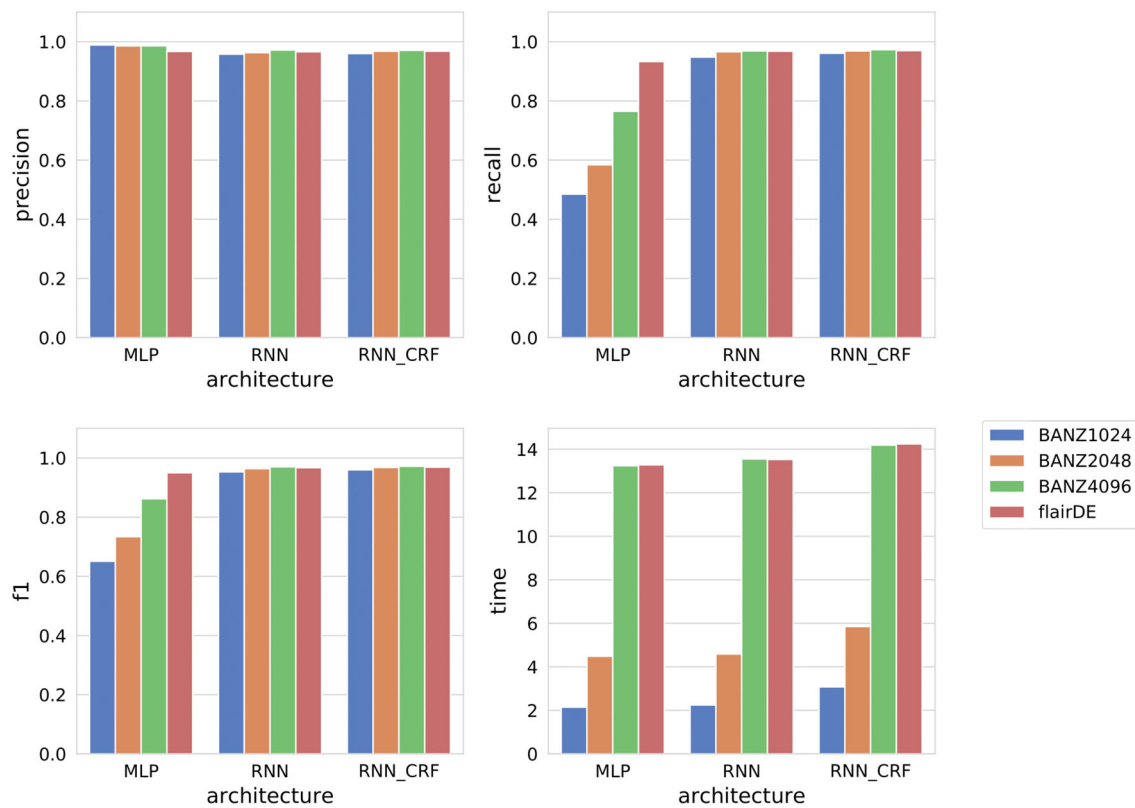


Fig. 4 Influence of language model on precision, recall, F₁-score and inference time on evaluation documents. Precision and recall are reported without post-processing. Inference time measured in seconds per document (10 pages). We see that for the RNN-based architectures, the choice of language model makes little difference in anonymiza-

tion performance. However, a smaller language model reduces the time it takes to process one document significantly. Note that there are no major differences in processing time between classifier architectures, the language model is the main contributor to processing time

BANZ2048, *BANZ4096*, respectively. We train for 100 epochs using the default parameters suggested by the package.

5.1.4 BERT language models

The used *BERT* model consists of a model pretrained on general German language data¹⁹ that we fine-tuned (i.e., continued to train) on the *BANZ* data corpus described above. The model provides embeddings of dimension 3072.

5.1.5 Classifiers

The RNN classifier as suggested by [14] is a one-layer BiLSTM with a hidden representation of 256 dimensions. We use the framework provided by the *flair* package to train RNN-based NER classifiers on the NER training dataset. We train for 100 epochs using the default parameters suggested by the package. Each MLP model consists of one intermediate hidden layer, mapping the input onto a lower-dimensional

representation. This hidden representation is then mapped onto the 7-dimensional output vector. The number of neurons in the intermediate hidden layer are 500, 500, 1000, depending on the input dimension 1024, 2048 and 4096, respectively. We train the MLP classifier for 100 epochs, using a batch size of 100 tokens. As optimizer, we use Adadelta with a learning rate of 0.1 and weight decay of $1e-5$.

Further, to provide a baseline evaluation we consider a pre-trained classifier for named entity recognition that has been trained on general language and named entity recognition data and has never seen our *BANZ* corpus or any annotated financial documents. For this, we apply the pre-trained NER model provided by the *flair* package, which is a RNN+CRF classifier trained on the CoNLL-2003 German NER dataset [13] and a general corpus language model. We denote this classifier as *flairNER*.

¹⁹ <https://deepset.ai/german-bert>.

5.2 Results and analysis

In this section, we present quantitative results on the performance of the described language models and classifiers. For our task of anonymization, it is desired to have a good binary classification performance, i.e., we tolerate a *PER* entity being tagged as an *ORG* entity and at the same time, we consider a *PER* entity tagged as *0* as a mis-classification and vice versa. For this reason, before evaluation all predicted and annotated tags are re-mapped onto two classes only, the negative class *0* indicating they are not sensitive entities and the positive class *1* indicating such tokens to be anonymized. Further, we are mostly interested in the performance on the positive class and therefore provide its metrics (precision, recall and F₁-score) only. Due to the lack of reliable available data for *SEG*, *PROD* and *OTH*, we do not consider them during this evaluation.

Table 1 presents the complete experimental results with different classifier architectures and language models. The evaluation on financial documents suggests that the RNN+CRF achieves the best performance, at over 97% recall without post-processing and around 99% after post-processing, without compromising precision of over 90%.

This results in a near complete anonymization of the entire document with very little unnecessarily anonymized words. Using domain-specific language model gives slight improvements over general language models for RNN-based classifiers. On the other-hand, the general corpus was beneficial while using a MLP classifier.

Figure 4 captures the influence of language model on the performance metrics. From the runtime and recall plots, we can observe that even with the smaller domain-specific language models, the RNN classifiers are able to out-perform the general language model, while reducing the runtimes of the anonymization process by over 50%. We further see that the RNN-based prediction models achieve comparable results for the larger RNN-based language models and the transformer-based *BERT*. Depending on application a slight drop in recall when employing a smaller language model (e.g., going from *BANZ4096* to *BANZ2048*) can be tolerated, considering it greatly improves on the inference time per document.

In order to evaluate the generalizability of our classifiers, we evaluate our models on GermEval dataset. For this evaluation, we do not apply any post-processing since it contains only sentences obtained from different sources and they do not follow any document structure. The results suggest that the RNN classifiers using a general language model performs better than one trained only on financial documents, which is expected since the sentences in GermEval correspond to sentences from a variety of sources. Nevertheless, the performance is comparable to the current state-of-the-art for NER. Further, the pre-trained NER classifier, trained on a general

language German NER corpus, only yields a recall of 84% and 93% on the financial documents, without and with post-processing, respectively.

6 Discussion and future work

In this work, we focus on the anonymization of financial documents and mention the use case for court records and legal documents in general. Another example for a possible application is healthcare. The ongoing battle with the corona pandemic showed how beneficial it is when hospitals and researchers work together and share their findings and information. At the same time, patient data often contains sensitive information prohibiting a fast exchange without prior anonymization. Therefore, an expansion of our approach to this field can enable and speed-up the data transfer and increase the amount of available data.

In order to apply this work to a new group of documents, one can use the following approach. As there are many similarities between entities of different domains, the presented models will likely work well even with no adaption. As seen in Table 1, a model pre-trained on general text data already performs decently at almost 90% anonymization performance. The next step to further increase the performance and recognize new patterns will be to train a domain specific language model and if available, fine-tune the model on annotated data of that field. We expect the post-processing steps described in Sect. 4.3 to also improve anonymization in most other domains, domain-specific post-processing steps might have to be developed.

In the experiments, we consider *BERT* as a contextualized language model that provides word embeddings which are passed as inputs to the separate prediction model. To further improve the language model, we plan on integrating named entities directly into the pre-training. Yamada et al. [24] show that treating words and entities as independent tokens during the masking task and within the self-attention mechanism can lead to better performances on named entity recognition tasks. Furthermore, we intend to explore transformers and self-attention as an end-to-end model for named entity recognition.

Another limiting factor for our method that inspires further research is the quality of annotations. Often, mistakes in the annotation lead to worse models by internalizing annotation mistakes during training. Additionally, Manning [25] demonstrates that the agreement between annotators can be another constraint. In the future, we intend to reduce the effect of both cases by identifying suspicious samples during training as shown by [26].

7 Conclusion

We presented a method to reliably anonymize the names of persons, locations and organizations using state-of-the-art deep learning techniques, as well as URLs, telephone numbers, dates and other numbers using classical rule-based approaches in financial documents. For internal use, this method can be applied to a single document or entire document corpora using a web-based application and a REST API. This allows for pre-processing of documents that can then be used by developers and researchers to train and evaluate further models for machine learning on financial data (e.g., [27]).

A quantitative evaluation of language models and text classifiers shows that domain-specific training of language models improves classification performance and smaller language models significantly improve runtime while maintaining anonymization performance. As future work, we would like to incorporate methods to anonymize additional identifying information (e.g., the segments the organization operates in) as well as analyze the impact of anonymized data as inputs for the training of machine learning algorithms over the original text.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s41060-021-00285-x>.

Funding Open Access funding enabled and organized by Projekt DEAL. Part of the authors from Fraunhofer IAIS were supported in parts by the Fraunhofer Research Center for Machine Learning (RCML) within the Fraunhofer Cluster of Excellence Cognitive Internet Technologies (CCIT) and by the Competence Center for Machine Learning Rhine Ruhr (ML2R) which is funded by the Federal Ministry of Education and Research of Germany (Grant No. 01IS18038B). We gratefully acknowledge this support.

Data availability Data for language modeling training is available at <https://www.bundesanzeiger.de/>. Annotated data for prediction model training is not publicly available.

Code Availability Code is not publicly available.

Declarations

Conflict of interest None.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the

permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Gola, P., Eichler, C., Franck, L., et al.: Datenschutzgrundverordnung: Ds-gvo (2017). Art. 6, paragraph 255
- Sweeney, L.: Replacing Personally-Identifying Information in Medical Records, the Scrub System. (1996)
- Wellner, B., Huyck, M., Mardis, S., et al.: Rapidly retargetable approaches to de-identification in medical records. *J. Am. Med. Inf. Assoc.* **14**(5), 564 (2007)
- Gardner, J., Xiong, L.: HIDE: an integrated system for health information de-identification. In: Proc. on. International Symposium on Computer-Based Medical Systems (2008), pp. 254–259
- Neamatullah, I., Douglass, M.M., Li-wei, H.L., et al.: Automated de-identification of free-text medical records. *BMC Med. Inf. Decis. Mak.* **8**(1), 32 (2008)
- Ferrández, O., South, B., Shen, S., et al.: BoB, a best-of-breed automated text de-identification system for VHA clinical documents. *J. Am. Med. Inf. Assoc.* **20**(1), 77 (2012)
- Nguyen, N., Guo, Y.: Comparisons of sequence labeling algorithms and extensions. In: Proceedings of the 24th International Conference on Machine Learning **227**, 681–688 (2007)
- Li, J., Sun, A., Han, J., Li, C.: A Survey on Deep Learning for Named Entity Recognition (2018)
- Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient Estimation of Word Representations in Vector Space (2013)
- Pennington, J., Socher, R., Manning, C.D.: GloVe: Global vectors for word representation. In: Proc. of Empirical Methods in Natural Language Processing (2014), pp. 1532–1543
- Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing (2018)
- Ghannay, S., Favre, B., Estève, Y., Camelin, N.: Word Embeddings Evaluation and Combination. Tech. rep. <https://code.google.com/p/word2vec/>
- Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Daelemans, W., Osborne, M. (eds.) Proc. of CoNLL, pp. 142–147 (2003)
- Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: Proc. of Int. Con. on Computational Linguistics (2018), pp. 1638–1649. <https://www.aclweb.org/anthology/C18-1139>
- Ramamurthy, R., Stenzel, R., Sifa, R., Ladi, A., Bauckhage, C.: Echo state networks for named entity recognition. In: Proc. of International Conference on Artificial Neural Networks (2019)
- Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of Tricks for Efficient Text Classification. arXiv preprint [arXiv:1607.01759](https://arxiv.org/abs/1607.01759) (2016)
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need, CoRR. arXiv preprint [arXiv:1706.03762](https://arxiv.org/abs/1706.03762)
- Bai, S., Koltzer, J.Z., Koltun, V.: An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, CoRR. arXiv preprint [arXiv:1803.01271](https://arxiv.org/abs/1803.01271)
- Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, CoRR. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805)
- Dai, Z., Yang, Z., Yang, Y., Carbonell, J.G., Le, Q.V., Salakhutdinov, R.: Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context, CoRR. arXiv preprint [arXiv:1901.02860](https://arxiv.org/abs/1901.02860)

21. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**, 1735 (1997)
22. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: *Proc. of Int. Conf. on Machine Learning (2001), ICML '01*, pp. 282–289
23. Benikova, D., Biemann, C., Kisselew, M., Padó, S.: GermEval Named Entity Recognition: Companion paper, pp. 104–112. In: *Proc. of the KONVENS GermEval Shared Task on Named Entity Recognition*, Hildesheim, Germany (2014)
24. Yamada, I., Asai, A., Shindo, H., Takeda, H., Matsumoto, Y.: LUKE: Deep Contextualized Entity Representations with Entity-Aware Self-Attention. *arXiv preprint [arXiv:2010.01057](https://arxiv.org/abs/2010.01057)* (2020)
25. Manning, C.D.: Part-of-speech tagging from 97 to 100%: is it time for some linguistics?. In: *International Conference on Intelligent Text Processing and Computational Linguistics (Springer, 2011)*, pp. 171–189
26. Wang, Z., Shang, J., Liu, L., Lu, L., Liu, J., Han, J.: Crossweigh: Training Named Entity Tagger from Imperfect Annotations. *arXiv preprint [arXiv:1909.01441](https://arxiv.org/abs/1909.01441)* (2019)
27. Sifa, R., Lübbering, M., Nütten, U., Bauckhage, C., Warning, U., Fürst, B., Khameneh, T., Thom, D., Huseynov, I., Kahlert, R., Schlums, J., Ladi, A., Ismail, H., Kliem, B., Loitz, R., Pielka, M., Ramamurthy, R., Hillebrand, L., Kirsch, B., Bell, T.: Towards automated auditing with machine learning. *Proc. ACM Symp. Doc. Eng.* **2019**, 1–4 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.