Iterative Named Entity Recognition with Conditional Random Fields

Ana Alves-Pinto¹,⁎, Christoph Demus²,†, Michael Spranger²,†, Dirk Labudde² and Eleanor Hobley¹,†

Abstract: Named entity recognition (NER) constitutes an important step in the processing of unstructured text content for the extraction of information as well as for the computer-supported analysis of large amounts of digital data via machine learning methods. However, NER often relies on domain-specific knowledge, being conducted manually in a time- and human-resource-intensive process. These can be reduced with statistical models performing NER automatically. The current work investigates whether Conditional Random Fields (CRF) can be efficiently trained for NER in German texts, by means of an iterative procedure combining self-learning with a manual annotation–active learning–component. The training dataset increases continuously with the iterative procedure. Whilst self-learning did not markedly improve the performance of the CRF for NER, the manual annotation of sentences with the lowest probability of correct prediction clearly improved the model F1-score and simultaneously reduced the amount of manual annotation required to train the model. A model with an F1-score of 0.885 was able to be trained in 11.4 h.

Keywords: active learning; self-learning; text; annotation; language

1. Introduction

The analysis of large amounts of unstructured text information can be supported and sped up by machine-learning-based methods. The successful use of such methods requires, however, the training of underlying machine learning models with accurately annotated or labelled text datasets [1]—that is, datasets containing metadata information on predefined information units. Model training needs to be conducted on domain-specific information or datasets that are appropriate to the information needs. For numerous applications regarding the identification and annotation of information units or entities such as persons, organisations, or location names in texts, a process called named entity recognition (NER) is highly relevant [2]. Accurate labelling of such entities is generally achieved by performing the annotation manually, because domain-specific knowledge is often required [3]—for example, to annotate medical terms across large amounts of medical records. Such manual annotation processing is, however, time-intensive, requiring a significant amount of human work. This becomes even more critical when analysing and annotating large numbers of text documents. Similar to other domains, the security sector also faces the challenge of analysing increasing amounts of heterogeneous information in an unstructured form.

Reduction of the time and effort required to label text datasets has been sought by employing machine-learning-based methods to perform NER automatically. Comprehensive reviews on NER discussing traditional approaches as well as machine-learning- and deep-learning-based approaches for NER can be found in [2,4]. State-of-the-art models to perform supervised NER include Conditional Random Fields (CRF) [3,5–7] and Long...
Short-Term Memory Networks (LSTMs) [7–10]. These have been reported to achieve good performance for English news texts (e.g., F1 Score of 0.9; [8–10]) but to perform poorer for German texts [11,12]. Ref. [7] trained and tested CRFs and Bilateral LSTMs (BiLSTM) on a dataset composed of German legal documents. They reported that BiLSTMs achieved superior performance with F1-scores of 95.95 for coarse-grained NER classes, compared with the 93.23 achieved by CRFs; further, they reported on the effect of unbalanced entity representations on model performance.

In addition to this, and despite the good performance reported, these models still need to be trained with manually labelled datasets in a supervised approach, with manual labelling remaining important due to domain specificities. This means that they only partially solve the need for reducing manual labour. In order to address this issue, previous works have complemented the above models with self-learning and or active learning approaches [13,14]. The former starts with an initial small set of manually labelled sentences for training and extends this set iteratively with sentences that are automatically labelled by the model in an unsupervised way. Generally, sentences with a high likelihood of a correct annotation are added. For instance, for probabilistic models such as CRFs, sentences are added with a high probability of correct classification of entities by the model [14,15]. Implementation of self-training, however, depends on several parameters, and generalisation of the trained models is not given [16]. In active learning, sentences are chosen from the pool of data for manual annotation before being added to the set of sentences for training. Sentences are selected by the model such as to maximise information learning by the model and minimise the amount of manual annotation required for training. This approach has been successfully employed by Tran and colleagues on Twitter posts in English [17].

The extent to which this method can contribute to substantially reduce the manual annotation of texts in other domains or languages (e.g., German) remains unclear, since the language usage and diversity of information that are dealt with are very different from that of Twitter posts. It is also unclear as to the extent to which these methods may support the recognition of unrepresented entities, as reported by [7].

In the current work, we investigated the extent to which a CRF-based algorithm in combination with active learning and self-training contributes to more efficient processing and consequent analysis of information contained in large amounts of German texts in a given domain. A further advantage is that this approach is independent of the domain or of the specific entities analysed, and is therefore applicable to different contexts and use cases. Despite the higher performance of BiLSTMs relative to CRFs reported previously, the latter was chosen to investigate the contribution of active and self-learning to the annotation effort of datasets since, as probabilistic and discriminative models, CRFs retrieve the probability of a possible labelling sequence for a given sequence of tokens [18,19], providing a means for selecting instances for active and self-learning. The CRF output probabilities are computed based on the syntactic structure and on observed variables for the token sequence given. By modelling probabilities, CRFs make iterative training of the model possible with a procedure encompassing a self-learning (i.e., unsupervised) component that supports an active learning (i.e., supervised) component via manual labelling of entities. In order to reduce the share of manual labelling required to train the model, it would be desirable/advantageous for the self-learning component to be the main contributor to the training of the model, i.e., to the improvement of the model performance. The influence of these two components on model training remains an open research question.

Investigating the contribution of these components on the training of a CRF model for NER was aimed not only at assessing the pros and cons of both learning approaches, but also to investigate the extent to which the share of manual labelling can be reduced and replaced by training via unsupervised learning (that is, self-training). For this purpose, we trained several CRF models, each with different implementations of active and self-learning, and compared their performance for NER. In addition to this, we estimated the efficiency of such a procedure by estimating the fraction of time required for manual and automatic
labelling separately. The latter includes the computation time for self-learning and for CRF training.

2. Materials and Methods

2.1. Dataset

The dataset is composed by 1082 text documents, encompassing 60,000 sentences and 1,567,625 tokens, all in German language. The dataset was self-generated by downloading newspaper articles and documents from the internet, all freely available at the time of download. Economy and finance subjects were chosen in terms of domain, since, in relation to a dataset composed of twitter posts for instance, it allowed the use of a relatively wide range of types of documents and formal language use. The dataset contains articles from online newspapers on financial areas (31% of the total number of tokens in the dataset), to reports from different financial institutions (31% of the dataset tokens), bulletins of governmental institutions (similarly, 31%), and press releases and other types of documents (about 7%). Web crawling of websites was performed with the Python tool Beautiful Soup https://www.crummy.com/software/BeautifulSoup/ accessed on 1 March 2020.

2.2. Data Preprocessing

The content of the original documents—in .HTML, .pdf, and .txt formats—was first converted in .txt format before being processed with the standard natural language processing (NLP) pipeline: sentence splitting, tokenization, and classification of tokens (e.g., symbol, regular, number), which was performed with the SoMaJo tool [20], whilst Part-Of-Speech tagging (POS-tagging) was conducted with the SoMeWeTa tool [21] and considering the German Newspaper Model. Lemmatization was also performed, for which the GermaLemma [22] tool specific for the German language was used. For tokens for which no lemma can be defined (for example, proper names or punctuation marks), the token itself was assigned as lemma (Figure 1).

![Preprocessing and NER modelling of the text dataset with indication of the tools used in each experimental step.](image-url)
2.3. Training, Optimization and Evaluation Datasets

The dataset was divided document-wise into a (1) training, (2) an optimisation, and (3) an evaluation dataset, each containing approximately 80%, 10%, and 10% of the sentences, respectively. The training dataset was used for model training purposes only. The optimisation set was employed on each iteration to select the optimal model parameters, whilst the evaluation set formed the independent set of data used for the final evaluation of model performance. The two latter datasets were completely manually annotated, for which we used the tool INCEpTION [23]. Labels consisted of the seven entities described in the sixth Message Understanding Conference (MUC-6) [24]. The three datasets have a similar composition in terms of document types, for example, the proportion of reports and newspaper articles. This similarity is reflected in the proportion of tokens belonging to each entity class and in the proportion of annotated tokens, which are similar for both the Optimization and Evaluation datasets (Table 1). The number of tokens, instead of the number of entities belonging to a class, was considered here since these two variables can, in several cases, be different; for example, in the case of the German federal state Nordrhein-Westfalen. Furthermore, there are tokens that can belong simultaneously to different entity classes; for example, the term Germany is not only a location (LOC) but, depending on the context, can also refer to a governmental institution (ORG).

Table 1. Number of tokens belonging to each of the seven entities labelled and the associated proportion relative to the total number of entities detected, for the Optimization and Evaluation datasets that were manually annotated. The last row indicates the total number of tokens belonging to any of the seven entities considered (‘#Entities’) as well as the proportion of the annotated tokens (independent of entity class) relative to the total number of tokens in the dataset (‘#Tokens’).

| Entity | Optimization Dataset | | Evaluation Dataset | | Total |
|---|---|---|---|---|---|
| | #Tokens | Proportion (%) | #Tokens | Proportion (%) | #Tokens | Proportion (%) |
| DAT | 4068 | 29.70 | 4080 | 34.20 | 8976 | 32.00 |
| ORG | 3509 | 25.70 | 4020 | 28.00 | 7529 | 26.90 |
| MON | 3015 | 22.00 | 2701 | 18.80 | 576 | 20.40 |
| LOC | 1131 | 8.30 | 1050 | 7.30 | 2181 | 7.80 |
| PCT | 1097 | 8.00 | 993 | 6.90 | 2090 | 7.50 |
| PER | 836 | 6.10 | 664 | 4.60 | 1500 | 5.40 |
| TIM | 23 | 0.20 | 12 | 0.10 | 35 | 0.10 |
| #Entities | 13,679 | 9.10 | 14,348 | 9.20 | 28,027 | 9.10 |
| #Tokens | 150,745 | | 156,278 | | 307,023 | |

2.4. Iterative Procedure for Annotation

The iterative procedure started with the manual annotation of a small set of 500 sentences randomly selected from the sentences belonging to the training dataset (Figure 2). This set of sentences contained 12,954 tokens, 959 of which were labelled as belonging to an entity. This set constituted the first ‘Model-Training dataset’ used, during the ‘Training and Prediction’ part of the procedure, to train an initial CRF model.

The initially trained model was then employed to classify entities in a pool of sentences randomly selected from the training dataset. This pool of sentences, newly generated during each iteration, contained 10x the number of sentences to be added to the ‘model-training dataset’ per iteration. In case not enough sentences were available for annotation in the training dataset, the pool-size was automatically reduced, with all sentences of the pool being added to the ‘Model-training dataset’. It should be noticed that a given sentence was either part of the initial model training dataset, part of the pool, or part of the set of unlabelled sentences left in the training dataset; these three groups of data never superposed during the iterative procedure.

After the ‘Training and Prediction’ step, new data were annotated either manually or automatically in the ‘Analysis and Annotation’ step before being added to the model training data. The aim of this step was, firstly, to identify sentences with the highest probability of correct classification by the model. In each iteration, these ‘sure sentences’ were automatically
annotated for entities by the model and added to the model training dataset without a manual check (self-training). Secondly, ‘unsure sentences’ with a low probability of correct classification for potential entities contained within them were identified by the model and presented for manual annotation in each iteration. This iterative manual annotation and expansion of the model training dataset represents the active learning component of the model training procedure. The prediction of the uncertainties of the correct classification was based upon the probabilities delivered by the CRF as a probabilistic model. The ‘sure sentences’ were those sentences with the highest probability of correct classification by the model. The ‘unsure sentences’ came from the lower quantiles of the probability distribution of the uncertainties provided by the model. During the automatic, self-training component of data annotation for the model training dataset, two types of sentences were distinguished—those for which at least one entity was predicted and those without predicted entities. To avoid a shift in the model training dataset to ‘sure’ sentences without entities, the automatically annotated self-training data added was set to be proportional to the number of sentences with at least one entity in the initially annotated training dataset, such that 48.8% of the sentences in the initially annotated data contained entities.

Figure 2. Schematic description of the sequence of steps composing the iterative procedure investigated.

This procedure of ‘training and prediction’ and ‘analysis and annotation’ was implemented iteratively to expand the annotated data as a basis for iterative model training with the overall goal of efficiently producing a model with high accuracy for NER. As such, the size of the model training dataset grew and the size of the training dataset shrank during each iteration. Rather than train the algorithm until no more sentences were left in the training dataset, we stopped the iterative training procedure after confirmation of convergence of the model performance, i.e., after no visible increase in performance during the previous iterations was observed (see Section 3).

2.5. Model Training and Fitting

Different token-associated variables and properties determined during preprocessing (Table 2) were used as input to the CRF model during training to recognise entities in the corpus. Feature-based analyses were restricted to trigrams—that is, features were considered for the token in question as well as for the single tokens immediately next to it. For tokens at the beginning or end of sentences, only bigrams were used; the position of these tokens was coded with BOS/EOS variables (Table 2). In addition to these features, the
following gazetteers were also employed for NER training: a list of all countries, a list with
German first names, and a list of date-related expressions (i.e., days, months, and seasons).

We employed the CRFsuite [25] to implement a linear-chain (first-order Markov) CRF
model in Python. The Limited-Memory-BFGS [26] algorithm was used to train the model,
with the maximum number of iterations set to 100 and allowing for all possible transition
features. L1 and L2 regularisation was applied to improve micro-F1-based model quality
(see Section 2.7.1). Parameter selection was based on a parameter grid with L1 and L2
varying in the intervals [0.01, 0.02] and [0.2, 0.8], respectively. These intervals were based
on results from preliminary model runs.

Table 2. Set of token semantic and syntactic features used in the CRF model.

| Variable      | Type of Data | Description                                           |
|---------------|--------------|-----------------------------------------------------------|
| word.lower    | String       | Token in lowercase letters                          |
| lemma         | String       | Token’s lemma                                      |
| word.isupper  | boolean      | All letters are uppercase                          |
| word.isdigit  | boolean      | All characters are numbers                          |
| word[-3:]     | String       | Token’s last 3 characters                           |
| word[-2:]     | String       | Token’s last 2 characters                           |
| word[ :3]     | String       | Token’s first 3 characters                          |
| word[ :2]     | String       | Token’s first 2 characters                          |
| pos_fine      | String       | Stuttgart–Tuebinger Tagset (STTS-Tagset)           |
| pos_coarse    | String       | POS Tag of the main word                           |
| class         | String       | Token-class (regular, symbol, numeric,...)         |
| BOS           | boolean      | Begin of sentence                                  |
| EOS           | boolean      | End of sentence                                    |
| isStopword    | boolean      | Token included in list of stopwords                |
| shape         | String       | Replacement of classes of characters: lowercase -> ‘a’,
                | upppercase -> ‘A’, characters -> ‘n’, Symbols -> ‘s’ |
| short_shape   | String       | replacement of a sequence of equal characters in ‘shape’ by one single character |
| isState       | boolean      | Token included in the list of countries            |
| isPrenome     | boolean      | Token included in the list of proper names         |
| isDateExpr    | boolean      | Token included in the list of date-related expressions |

2.6. Experimental Design and Nomenclature

In order to answer our research questions relating to the influence of self-training-
and active-learning-supported manual annotation on model performance as well as the
time required to train an accurate model, we performed different experiments. In these,
we focused on investigating the influence of the probability distribution used for selection
of data for manual annotation and the number of data points manually annotated during
each iteration, as we hypothesised that these factors would have the largest influence on
information gain for the model and the time needed to train a good model. Additionally, we
varied the size of the pool for annotation and investigated its effect on model performance.

2.6.1. Reference Experiment

To start with, we performed a reference experiment, which formed the basis for
comparison with the subsequent experiments. This reference experiment was performed
with self-training and active learning components. In this experiment, the number of
automatically annotated data points during self-training was set to one tenth the number of
data points in the model training data per iteration. Forty ‘unsure’ sentences were
manually annotated during each iteration and added to the model training dataset. Twenty
of these sentences were selected from the lowest quantile of the probability distribution
of the uncertainties obtained by the model, and twenty sentences were selected from the
0.2 quantile upwards (‘Ann. 20-0Q, 20-0.2Q, ST’, Table 3).
Table 3. Description of the different experiments conducted to evaluate the effect of self-training, the number and selection of sentences for manual annotation, and the pool size of the model training for NER.

| Experiment Designation | Description |
|------------------------|-------------|
| Ann. 20-0Q, 20-0.2Q; ST | Manual annotation of 20 sentences selected from the lowest quantile (0-Q) plus 20 sentences from the 0.2-quantile upwards; self-training (ST). This constitutes the reference experiment. |
| no Ann.; ST | No iterative manual annotation, apart from the initial annotation; self-training (ST) only. |
| Ann. 20-0Q, 20-0.2Q; no ST | Manual annotation of 20 sentences from the lowest quantile (0-Q) and 20 from the 0.2-quantile upwards (20-0.2Q); no self-training (no ST). |
| Ann. 20-0Q, 20-0.2Q; no ST; small pool size | Manual annotation of 20 sentences from the lowest quantile (0-Q) plus from the 0.2-quantile (0.2-Q) upwards; no self-training; Pool size set to 10 times the number of manually annotated sentences, i.e., 400 sentences. |
| Ann. 20-0Q, 20-0.2Q; no ST; large pool size | Manual annotation of the 20 sentences from the lowest quantile (0-Q) and 20 from the 0.2-quantile (0.2-Q) upwards; no self-training (no ST); pool size equal to that used in the equivalent experiment with self-training (see first row). |
| Ann. 40-0Q; no ST | Manual annotation of 40 sentences from the lowest quantile (0-Q); no self-training (no ST). |
| Ann. 40-random; no ST | Manual annotation of 40 sentences selected randomly from the pool; no self-training (no ST). |
| Ann. 40-0.2Q; no ST | Manual annotation of 40 sentences from the 0.2-quantile upwards; no self-training (no ST). |
| Ann. 20-0Q; no ST | Manual annotation of 20 sentences from the lowest quantile (0-Q); no self-training (no ST). |

2.6.2. Influence of Self-Training on Model Performance

To assess the influence of self-training on model performance, we performed an experiment without manual annotation using only the self-training component in each iteration. In this experiment, the ‘sure’ sentences from the pool were selected for automatic annotation and added to the model training data during each iteration (‘no Ann., ST’, Table 3). The number of automatically annotated data points during self-training was set to one tenth the number of data points in the model training data per iteration. This was contrasted with an experiment in which we only applied the active learning and manual annotation component without self-training. The manually annotated sentences were composed of twenty sentences selected from the lowest quantile of the probability distribution of the uncertainties obtained by the model, and twenty sentences from the 0.2-quantile upwards (‘Ann. 20-0Q, 20-0.2Q, no ST’, Table 3).

In the experiments with only self-training or active learning, the size of the pool was comparatively smaller than in experiments with both components, as less sentences were added to the model training data in each iteration step. This was particularly the case for the
experiment without self-training compared with the experiment with self-training, as the majority of added sentences in each iteration were derived from the self-training component during model training. In order to assess the effects of the comparative reduction in the model training dataset as a result of the omission of self-training on the model performance, we performed an experiment without self-training and with an adjusted pool size. The pool size in this experiment was set to the pool size in a comparable experiment with self-training, but only manual annotation was performed (‘Ann. 20-0Q, 20-0.2Q, no ST, large pool-size’, Table 3). In a further experiment, we retained the pool size at a constant 400 sentences (i.e., 10 times the number of sentences annotated and added to the model training data) per iteration (‘Ann. 20-0Q, 20-0.2Q, no ST, small pool-size’, Table 3).

2.6.3. Influence of Active-Learning-Supported Manual Annotation on Model Performance

In order to assess the effects of active learning on model performance, we performed several experiments training the algorithm using only manual annotation but without self-training for data annotation whilst varying the number of data points for annotation and the probabilities used to select the data points for annotation. In three experiments, we annotated 40 sentences manually per iteration, with the sentences being selected from either the lowest quantile (0Q), the 20% quantile (0.2Q), or an equal mixture of both upwards (‘Ann. 40-0Q, no ST'; 'Ann. 40-0.2Q, no ST'; 'Ann. 20-0Q, 20-0.2Q, no ST', Table 3) and added to the model training dataset. In a further experiment, we randomly selected 40 sentences for manual annotation in each iteration and added them to the model training dataset (‘Ann 40-random, no ST', Table 3). Lastly, we performed one experiment in which we annotated only the 20 ‘unsurest’ sentences from the lowest quantile for addition to the model training dataset in each iteration (‘Ann. 20-0Q no ST', Table 3).

2.7. Evaluation

2.7.1. Model Performance

We validated the model after each iteration using the optimisation dataset to assess the optimal L1 and L2 regularisation parameters (Section 2.5) and learning progress, and then trained an optimal model using these parameters. The final evaluation was performed using the independent evaluation dataset. Model performance was based on precision, recall, and the micro-F1-score as the harmonic mean between the microprecision and microrecall. Subsequently, when we refer to the F1-score, we imply the micro-F1-score. Additionally, we calculated and examined the F1-score for each entity individually.

As the pool is randomly selected in each iteration, each experiment has a random driver, so that via repetition of experiments, we were able to estimate the variance in model performance for each experimental setting. The variance was calculated by repeating each experiment five times with the same variables but using a different random seed in each repetition and estimating the variance in model performance between repetitions reported as the 95% confidence interval in model performance across repetitions.

2.7.2. Time Estimation for the Model Training Procedure

As not only the model performance but also the time required to annotate the data and train a model is highly relevant to practical applications of NER in specialised domains such as the security sector, we estimated the time to fully train the model $t_{\text{train}}$ for the most relevant experimental setting. This time interval needed to fully train the model is a function of both the annotation time for all annotated data points $t_{\text{ann}}$ and the calculation time for all iterations $t_{\text{calc}}$. The latter can be estimated from the average time required to fit the model per iteration $t_{\text{fit}}$ and the number of iterations $I$ per model:

$$t_{\text{train}} = t_{\text{ann}} + I t_{\text{fit}} = t_{\text{ann}} + t_{\text{calc}}$$

Both the annotation time and the average time to fit the model per iteration were estimated during the experiments. The former was linearly related to the number of manually annotated tokens in each experiment (Figure 3) and estimated at an average of
0.6 s per token. On the other hand, the latter depends on both the model training dataset, the optimal parameters, and the power of the CPU used to fit the model. The values here reported are based on modelling work that was performed using an Ubuntu-Linux OS V 18.04.1 with an Intel Core i7-8750H (6 core) processor and 32 GB RAM.

![Annotation Time Graph](image)

**Figure 3.** Estimation of the annotation time as a function of the number of tokens annotated.

### 3. Results

#### 3.1. Reference Experiment

After the first iteration, the model reached an F1-score of 0.786, with precision (0.878) higher than recall (0.711, Figure 4a). During the iterative training procedure, the recall increased strongly by 0.165 to a final score of 0.876, while the precision increased less dramatically to a final score of 0.929, resulting in a final F1-score of 0.902 after the 29th iteration of data annotation. After this iteration, the training was halted as the increase in model performance had plateaued. In total, 30 models (0th–29th iteration) were fitted during the procedure. The classwise F1-score per entity reveals high initial F1-scores of nearly 0.9 or above for the entities belonging to the NUMEX classes ‘percent’, ‘date’, and ‘money’ (Figure 4b), with minor improvements obtained after a relatively small number of annotated data points added to the model training dataset. The recognition of these entities was much better than that of entities belonging to the ENAMEX group (organisation, person, location), the initial F1-scores of which were all below 0.7, and whose improvement during iterative training was stronger than that obtained for the abovementioned entities. An exception were tokens classified as the entity ‘time’, which had an F1-score of <0.5 even after the annotation of 80,000 tokens, and the prediction of which only improved after the annotation of 100,000 tokens, yielding a final score of 0.8.

![Performance Graphs](image)

**Figure 4.** (a) Overall performance (F1-score) during the first 30 iterations during model training according to the settings of the reference experiment, i.e., with manual annotation of 20 sentences from the lowest quantile and 20 from the quantile 0.2 upwards (‘Ann. 20-0Q, 20-0.2Q; ST’ in Table 3). (b) Model performance for each separate entity (F1-score).
3.2. Influence of Self-Training on Model Performance

The experiment with self-training but without manual annotation revealed a clear trend to nearly no model improvement after even 18 iterations (Figure 5). This near-negligible contribution of the self-learning component of the algorithm to model performance was further demonstrated in the experiments performed with manual annotation and without self-learning (Figure 6), which revealed marginal differences in model performance compared with models trained with manual annotation and self-training with a large pool size (Figure 6a), with 95% CI nearly identical between experiments. For the experiment with the small pool-size without self-training, the difference between final F1-scores was larger, with the experiment without self-training performed on average better then with self-training, though the 95% CI clearly overlapped (Figure 6b).

Figure 5. Evolution of model performance (F1-score) with training when (1) only self-learning and no manual annotation was performed (red line) and (2) both self-learning and manual annotation were performed (blue line). The shadowed areas illustrate in each case the associated 95% confidence interval.

Figure 6. Influence of self-learning on model performance when a (a) large and (b) small pool size is used for training.

3.3. Influence of Sentence Selection for Manual Annotation on Model Performance

The mean model performance of the experiments without self-training using manual annotation of 40 sentences from the lowest quantile, or combined from the lowest quantile and the 20% quantile upwards, were nearly identical and did not significantly differ (Figure 7a). After manual annotation of nearly 73,000 tokens, the average F1-score was 0.885 and 0.887 for these experiments, respectively. The iterative annotation of 40 randomly selected sentences or 40 sentences from the 0.2 quantile upwards led to significantly poorer final models, with an F1-score of 0.02 less on average. The results of these two experiments
did not significantly differ, although the random sampling approach led to a larger 95% CI, indicating larger uncertainty in model results. Additionally, the latter two models required, on average, nearly 25,000 additional manually annotated tokens to achieve their final F1-score of 0.86 than the two former, better performing models, an increase of nearly 50% of the total number of annotated tokens.

The performance of the models based on annotation of the 20 ‘unsurest’ sentences per iteration, and based on 20 of the ‘unsurest’ sentences plus 20 sentences from the 0.2 quantile upwards, revealed nearly no differences up to the annotation of around 35,000 tokens (Figure 7b). However, with an increasing number of annotated tokens after, the model built using annotations of the 20 ‘unsurest’ sentences in each iteration performed better than the model with additional annotation of 20 sentences per iteration. The former required the annotation of approx. 18,000 tokens less to reach an F1-score of 0.885, the same as the final F1-score of the latter, reference experiment. Additionally, the model trained iteratively with manual annotation of the 20 ‘unsurest’ sentences had not plateaued after an average of 39 iterations (55,000 manually annotated tokens). In contrast, the reference model required, on average, 21 iterations for the manual annotation of 55,000 tokens and plateaued after on average 31 iterations and ca. 73,000 manually annotated tokens.

![Figure 7. Evolution of model performance (F1-score) with training iterations when (a) different sets of sentences are selected for manual annotation (see Table 3 for a description of the different experiments conducted). Self-learning was performed in none of the cases. The same is depicted in (b) but only two different selections are compared in this case.](image)

### 3.4. Estimated Time for Model Fit

The time required to fit a model with an F1-score of 0.885 (the best achieved with the reference experiment) was compared for three different training approaches (Table 4). The experiment with manual annotation of the 20 ‘unsurest’ sentences without self-training was fastest, requiring 11.4 h on average to achieve the stated F1-score. On average, the annotation time contributed around 80% of the total time for model fitting (Table 4). In the experiments in which 40 sentences were manually annotated per iteration, the annotation time was 2.2 to 3.6 h longer on average than in the experiments with annotation of 20 sentences per iteration. Although the former had lower calculation times, the overall time for model fitting was lowest in the experiment without self-training and with the manual annotation of merely 20 sentences. The experiment with the highest overall time was that which included self-training.
Table 4. Comparison of the overall time ($t_{\text{train}}$) required to train the model to an F1-score of 0.885, for three different training experiments. This overall time was estimated as the sum of the annotation time ($t_{\text{ann}}$) plus the calculation time ($t_{\text{calc}}$).

| Experiment                          | #Tokens | $t_{\text{ann}}$ (h) | Mean Number of Iterations | $t_{\text{calc}}$ (h) | $t_{\text{train}}$ (h) |
|-------------------------------------|---------|----------------------|---------------------------|----------------------|------------------------|
| 20-0Q; no ST                        | 54,157  | 9.2                  | 38.1                      | 2.2                  | 11.4                   |
| 20-0Q, 20-0.2Q, no ST                | 66,945  | 11.4                 | 29.1                      | 1.9                  | 13.3                   |
| 20-0Q, 20-0.2Q; ST                   | 75,231  | 12.8                 | 18.0                      | 1.4                  | 14.2                   |

4. Discussion

4.1. General Model Performance and Individual Entity Recognition

The initial strong improvement to the model performance of the reference experiment (and subsequent experiments) is attributable to the growing number of annotated entities used to fit the model during each iteration so that the model learns from more data. Additionally, the introduction of annotated, particularly ‘unsure’ sentences to the model training dataset provides an additional source of relevant model information in each iteration. The manually annotated sentences were particularly long, containing at times over 100 tokens per sentence, and encompassing a large number of entities. Through the annotation of many entities with high uncertainty, the model parameter space is expanded in regions lacking information in the prior iteration, enabling efficient information growth for the model in particularly uncertain regions in the parameter space. Thus, our results confirm previous findings highlighting the importance of data selection for model performance beyond the benefit of having large collections of documents, for instance, using bootstrapping procedures [27].

The comparatively lower performance of the model in recognising entities from the ENAMEX group can be attributed to the higher diversity of these entities in natural language usage than for entities from the NUMEX and TIMEX groups. NUMEX and TIMEX entities tend to have standard formats that can be well-estimated via regular expressions, whereas ENAMEX entities can be comprised of a vast, indeed nearly limitless, number of combinations of words, (invented) names, or acronyms. Thus, our results agree with previous reports of reduced vocabulary transfer of entities of the ENAMEX group, especially for persons and organisations [7,28,29]. Some regular expressions, such as the acronym GmbH for proprietary limited companies, can be used to identify entities in the German language, but not all entities can be identified using such expressions and not all texts relating to such an entity use the appropriate acronym. The poor performance of the reference experimental model at recognising the TIMEX entities after a large number of annotated tokens can be attributed to its rare occurrence in the data: in the evaluation dataset, only twelve instances were manually annotated.

4.2. Influence of Self-Training on Model Performance

The lack of improvement in model performance during the experiment with only self-training as well as by the comparison of model performance with and without ST indicates that self-training, as performed in our experimental setting, had no positive effect on model performance. This result contrasts with the studies of Tran et al. [17] and Steedman et al. [30], which indicated that self-training and automatic data annotation can work in principle. We attribute our divergent results to the large number of highly diverse ENAMEX entities in our dataset, which are more difficult for NER models to recognise. Furthermore, the sentences analysed were collected from different sources, including texts from newspapers and from technical reports. The resulting dataset is likely to have a larger diversity in linguistic style analysed relative to previous works—for example, from analysing Twitter posts—and this may also affect model performance [7]. However, more investigations are required to verify these hypotheses, and the effects of NLP models to recognise entities of different classes deserves more research. A possible research path involves the combination of CRFs with BiLSTMs, which can achieve state-of-the-art performance for specific language resources [10] and be potentially useful in the...
identification of entities of the ENAMEX group, e.g., [7,31]. In addition, methodological approaches involving latent factor analysis may also contribute to improve computational efficiency and prediction accuracy of trained models for unseen data [32–36].

In contrast to the lack of model improvement via (only) self-training, the marginal increase in model performance in the experiment without self-training and with a small pool size indicates that the model may learn better with the same number of annotated tokens. This is potentially associated with the higher number of iterations in these experiments, enabling more ‘unsure’ data points to be selected per iteration, driving better model performance.

4.3. Optimising Selection of Manually Annotated Sentences to Enhance Model Performance and Procedural Efficiency

The better performance of the models fitted with manually annotated data selected via active learning underscores our above results, indicating that annotation of ‘unsure’ sentences provides the most valuable information to the model. Although there was no statistical difference between the models based on manual annotation of sentences from the 0.2Q upwards or random sampling, the higher variance in model performance of the latter indicates that models based on random sampling of sentences are less reliable in their results than models created using uncertainty-based sampling in an active learning procedure. Compared with annotation of random sentences or sentences with a greater certainty estimated by the model, our results confirm that iterative manual annotation of highly uncertain sentences enhances final model performance and reliability as found in other studies [14].

The better performance of the model built with the manual annotation of a smaller number of highly uncertain data points per iteration can be attributed to an increase in the number of iterations performed when training the model. The reduction in manually annotated tokens per iteration leads to an increase in the total number of iterations, with the model seeing the least certain and, therefore, most informative data points in each iteration. This allows the model to learn more frequently and adapt in the most uncertain regions of the parameter space during each iteration, leading to an overall increase in model performance.

An additional benefit to selecting the most uncertain sentences for manual annotation was the large reduction in the number of tokens required to be annotated overall, which leads to greatly increased efficiency in the model fitting process performed in such an iterative procedure. Furthermore, the lack of a difference in model performance when annotating the 40 most uncertain sentences compared with 20 sentences sampled from the 0% quantile and 20 sentences sampled from the 20% quantile upwards indicates that, in our experimental setting, annotation of 20 sentences per iteration was sufficient to drive a good increase in model performance. In other words, the 20 most uncertain sentences contributed the most information for the model.

This result was underscored by the experiment with only manual annotation of the 20 ‘unsurest’ sentences, which required few annotations to achieve the same F1-score as the experiment with the iterative manual annotation of the 20 ‘unsurest’ sentences and 20 sentences from the 20% quantile upwards. The former had not plateaued at that point, indicating potential for improvement in model performance. Lastly, the experiment without self-training in which only the 20 ‘unsurest’ sentences were manually annotated per iteration required not only the least overall time for model fit, but importantly, required the least amount of time for manual annotation, indicating the greatest reduction in workload for the person developing the model.

5. Conclusions

Our examination of the influence of the self-training and active learning components of the conditional random fields algorithm implemented for named entity recognition revealed little to no value in self-training but a large positive effect on model performance as well as work efficiency of the active learning component. This positive effect of active
learning can be attributed to streamlined data selection and an increase in the number of model iterations, enabling the training of more accurate models with the annotation of fewer, more relevant data points, which are most helpful to model training when selected from the lowest quantiles of certainty obtained from the algorithm. The fastest training of a CRF model with an F1-score of 0.885 took 11.4 h and was achieved when, on each iteration, only 20 of the ‘unsurest’ sentences were manually annotated and no self-training was conducted. For the accurate recognition of ENAMEX-tokens—a central task for many NER application fields—we recommend further studies into active learning.

**Author Contributions:** Conceptualization, C.D., M.S., D.L. and E.H.; methodology, C.D., M.S. and E.H.; software, C.D.; validation, C.D.; formal analysis, C.D.; investigation, C.D.; resources, E.H.; data curation, A.A.-P. and C.D.; writing—original draft preparation, A.A.-P., C.D. and E.H.; writing—review and editing, A.A.-P.; visualization, A.A.-P. and C.D.; supervision, E.H.; project administration, E.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Restrictions apply to the availability of the data.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Pustejovsky, J.; Stubbs, A. Natural Language Annotation for Machine Learning: A Guide to Corpus-Building for Applications; O’Reilly Media, Inc.: Sebastopol, CA, USA, 2012; Volume 1.

2. Nadeau, D.; Sekine, S. A survey of named entity recognition and classification. *Lingvisticae Investig.* **2007**, *30*, 3–26. [CrossRef]

3. Sang, E.C.; De Meulder, F. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv* **2003**, arXiv:0306050.

4. Li, J.; Sun, A.; Han, J.; Li, C. A survey on deep learning for named entity recognition. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 50–70. [CrossRef]

5. Settles, B. Biomedical named entity recognition using conditional random fields and rich feature sets. In Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications (NLPBA/BioNLP), Geneva, Switzerland, 28–29 August 2004; Volume 4, pp. 107–110.

6. Okanohara, D.; Miyao, Y.; Tsuruoka, Y.; Tsujii, J. Improving the scalability of semi-markov conditional random fields for named entity recognition. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, 17–18 July 2006; Volume 5, pp. 465–472.

7. Leitner, E.; Rehm, G.; Moreno-Schneider, J. Fine-grained named entity recognition in legal documents. In Proceedings of the International Conference on Semantic Systems, Karlsruhe, Germany, 9–12 September 2019; Springer: Berlin/Heidelberg, Germany, 2019; Volume 9, pp. 272–287.

8. Chiu, J.P.; Nichols, E. Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 357–370. [CrossRef]

9. Huang, Z.; Xu, W.; Yu, K. Bidirectional LSTM-CRF models for sequence tagging. *arXiv* **2015**, arXiv:1508.01991.

10. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural architectures for named entity recognition. In Proceedings of the HLT-NAACL, San Diego, CA, USA, 12–17 June 2016; Volume 8.

11. Benikova, D.; Biemann, C.; Kisselew, M.; Pado, S. GermEval 2014 named entity recognition shared task: companion paper. In Proceedings of the KONVENS GermEval Workshop, Hildesheim, Germany, 8–10 October 2014; Volume 10, pp. 104–112.

12. Benikova, D.; Muhie, S.; Prabhakaran, Y.; Biemann, S.C. C.: Germaner: Free open german named entity recognition tool. In Proceedings of the GLC-2015, Duisburg, Germany, 30 September–2 October 2015; Volume 11.

13. Tsuruoka, Y.; Tsujii, J.; Ananiadou, S. Accelerating the annotation of sparse named entities by dynamic sentence selection. *BMC Bioinform.* **2008**, *9*, 1–10. [CrossRef]

14. Liao, W.; Veeramachaneni, S. A simple semi-supervised algorithm for named entity recognition. In Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing, Boulder, CO, USA, 5 June 2009; Volume 13, pp. 58–65.

15. Sammut, C.; Webb, G.I. Leave-one-out cross-validation. *Encycl. Mach. Learn.* **2010**, 600–601.

16. Kaljahi, R.S.Z. Adapting self-training for semantic role labeling. In Proceedings of the ACL 2010 Student Research Workshop, Uppsala, Sweden, 13 July 2010; Volume 15, pp. 91–96.

17. Tran, V.C.; Nguyen, N.T.; Fujita, H.; Hoang, D.T.; Hwang, D. A combination of active learning and self-learning for named entity recognition on twitter using conditional random fields. *Knowl.-Based Syst.* **2017**, *132*, 179–187. [CrossRef]
18. Sutton, C.; McCallum, A. An introduction to conditional random fields. Foundations and Trends® in. Mach. Learn. 2012, 4, 267–373.
19. Lafferty, J.; McCallum, A.; Pereira, F. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the ICML, Williamstown, MA, USA, 28 June–1 July 2001; Volume 18.
20. Proisl, T.; Uhrig, P. SoMaJo: State-of-the-art tokenization for German web and social media texts. In Proceedings of the 10th Web as Corpus Workshop, Berlin, Germany, 12 August 2016; Volume 19, pp. 57–62.
21. Proisl, T. SoMeWeTa: A part-of-speech tagger for German social media and web texts. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018; Volume 20.
22. Konrad, M. GermaLemma: A Lemmatizer for German Language Text; Berlin Social Science Center: Berlin, Germany, 2019.
23. Klie, J.C.; Bugert, M.; Boullosa, B.; de Castilho, R.E.; Gurevych, I. The inception platform: Machine-assisted and knowledge-oriented interactive annotation. In Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations, Santa Fe, NM, USA, 20–25 August 2018; Volume 22, pp. 5–9.
24. Grishman, R.; Sundheim, B.M. Message understanding conference-6: A brief history. In Proceedings of the COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics, Copenhagen, Denmark, 5–9 August 1996; Volume 23.
25. Okazaki, N. Crfsuite: A Fast Implementation of Conditional Random Fields. 2007. Available online: http://www.chokkan.org/software/crfsuite/ (accessed on 1 March 2020).
26. Nocedal, J. Updating quasi-Newton matrices with limited storage. Math. Comput. 1980, 35, 773–782. [CrossRef]
27. Ji, H.; Grishman, R. Data selection in semi-supervised learning for name tagging. In Proceedings of the Workshop on Information Extraction Beyond The Document, Sydney, Australia, 22 July 2006; Volume 26, pp. 48–55.
28. Palmer, D.D.; Day, D. A statistical profile of the named entity task. In Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, DC, USA, 31 March–3 April 1997; Volume 27, pp. 190–193.
29. Mikheev, A.; Moens, M.; Grover, C. Named entity recognition without gazetteers. In Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics, Bergen, Norway, 8–12 June 1999; Volume 28.
30. Steedman, M.; Osborne, M.; Sarkar, A.; Clark, S.; Hwa, R.; Hockenmaier, J.; Ruhlen, P.; Baker, S.; Crim, J. Bootstrapping statistical parsers from small datasets. In Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics, Budapest, Hungary, 12–17 April 2003; Volume 29.
31. Dias, M.; Boné, J.; Ferreira, J.C.; Ribeiro, R.; Maia, R. Named entity recognition for sensitive data discovery in Portuguese. Appl. Sci. 2020, 10, 2303. [CrossRef]
32. Wu, D.; Luo, X. Robust latent factor analysis for precise representation of high-dimensional and sparse data. IEEE/CAA J. Autom. Sin. 2020, 8, 796–805. [CrossRef]
33. Wu, D.; Luo, X.; Shang, M.; He, Y.; Wang, G.; Wu, X. A Data-Characteristic-Aware Latent Factor Model for Web Services QoS Prediction. IEEE Trans. Knowl. Data Eng. 2020. [CrossRef]
34. Luo, X.; Yuan, Y.; Chen, S.; Zeng, N.; Wang, Z. Position-transitional particle swarm optimization-incorporated latent factor analysis. IEEE Trans. Knowl. Data Eng. 2020. [CrossRef]
35. Luo, X.; Wu, H.; Wang, Z.; Wang, J.; Meng, D. A Novel Approach to Large-Scale Dynamically Weighted Directed Network Representation. IEEE Trans. Pattern Anal. Mach. Intell. 2021. [CrossRef] [PubMed]
36. Luo, X.; Zhou, B.; Liu, Z.; Zhou, M. Fast and Accurate Non-negative Latent Factor Analysis on High-dimensional and Sparse Matrices in Recommender Systems. IEEE Trans. Knowl. Data Eng. 2021. [CrossRef]