Named Entity Recognition for Specific Domains - Take Advantage of Transfer Learning

Sunna Torge, Waldemar Hahn, Lalith Manjunath, René Jäkel

***Abstract*—Automated text analysis as named entity recognition (NER) heavily relies on large amounts of high-quality training data. For domain-specific NER, transfer learning approaches aim at overcoming the problem of lacking domain-specific training data. In this paper, we investigate transfer learning approaches in order to improve domain-specific NER in low-ressource domains. The first part of the paper is dedicated to information transfer from known to unknown entities using BiLSTM-CRF neural networks, considering also the influence of varying training data size. In the second part instead, pre-trained BERT models are fine-tuned to domain-specific German NER. The performance of models of both architectures is compared w.r.t. different hyperparameters and a set of 16 entities. The experiments are based on the revised German SmartData Corpus, and a baseline model, trained on this corpus.**

***Index Terms*—Domain-specific Named Entity Recognition, BiLSTM-CRF, German, Transfer Learning, Transformers (BERT).**

I. Introduction

DOMAIN-SPECIFIC knowledge is increasingly hidden in large amounts of documents, which results in an urgent need for automated text analysis. However, a common obstacle is the lack of annotated domain-specific training data, that are expensive to collect and annotate, resulting in so-called low-resource domains. Transfer learning (TL) approaches may overcome this problem by transferring knowledge across different domains. The goal is to transfer knowledge from a related domain (called source domain) to improve the learning performance or minimize the number of labeled examples required in a target domain. Informally speaking, TL approaches enable the creation of a machine learning model for low-resource domains, based on the knowledge provided by an existing machine learning model, which has been trained on (a larger amount of) training data of a different domain.

This reuse of information also aims at reducing training times and at the same time obtaining the best descriptive or predictive performance. A thorough investigation of a large variety of TL approaches is represented in [1]. We define TL according to [1], [2] as follows.

Definition 1: (Domain) A domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$ consists of a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$. X denotes an instance set, defined as $X = \{\mathbf{x}|\mathbf{x_i} \in \mathcal{X}, i = 1, ..., n\}$.

Definition 2: (Task) A task $\mathcal{T} = \{\mathcal{Y}, f\}$ consists of a label space $\mathcal{Y}$ and a decision function f. The decision function f is an implicit one, which is expected to be learned from the sample data.

Definition 3: (Transfer Learning) Given a source domain $\mathcal{D}_S$ with a corresponding source task $\mathcal{T}_S$ and a target domain $\mathcal{D}_T$ with a corresponding task $\mathcal{T}_T$, transfer learning is the process of improving the target predictive function f_T by using the related information from $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. The case where the feature space $\mathcal{X}_S = \mathcal{X}_T$ is defined as homogeneous transfer learning.

In our work, we only consider homogeneous transfer learning. The task under consideration is Named Entity Recognition (NER), which is a classification task, assigning words or more complex parts of a text document (tokens) to predefined entity classes. Focusing on domain-specific German text data, we are constantly confronted with the lack of training data. This led us to the general question: which TL approaches could support NER on a domain-specific German corpus and how. Our investigations were motivated by the following three questions:

- Q1 Is it possible to transfer information within a German text corpus from one specific domain to another specific domain?
- Q2 What is the influence of the training data size on this special case?
- Q3 Does domain-specific NER also profit from general domain knowledge and general task knowledge as provided by BERT language models?

Our investigations are based on the SmartData corpus [3], published in 2018. The SmartData corpus was the only available German corpus covering two different specific domains, namely traffic and industry, and providing annotations for 16 (domain-specific) entities. This enabled us to address Q1 and Q2 with a *homogeneous* TL approach, based on a Bidirectional Long-Short-Term-Memory Neural Network with Conditional Random Fields (BiLSTM-CRF) as described in section IV. The first part of the experiments comprises of the source model training for only 15 entities, and uses this knowledge for the target model training for the remaining entity. Thus in this experiment, the set of labels in the source domain $\mathcal{Y}_S$ differs from those in the target domain $\mathcal{Y}_T$. These models are compared with models, trained without transfer learning. In the second part the impact of training data size is investigated. The results of these experiments (section V) are discussed in section VI.

Almost at the same time as the SmartData corpus, the new language representational model BERT (Bidirectional Encoder Representations from Transformers) was published [4], introducing a different concept of TL for down-stream tasks as NER. BERT models are trained using two unsupervised tasks, namely the masked language model (MLM) task and the next sentence prediction (NSP) task. The training is performed on

large general domain text corpora. We address Q3 by fine-tuning BERT models on the downstream task, NER, using the domain-specific German text corpus. Thus, again the transfer learning approach comprises of different tasks. In addition, the set of instances, X, in the source domain is enhanced with the SmartData corpus in the target domain. For fine-tuning, we used the German Bert model [5], which is the first BERT model being trained on German text data, and the multi-lingual BERT model, released by Google [6]. Training for refinement also includes a hyperparameter study. The resulting BERT models are compared with the previously obtained baseline BiLSTM-CRF model. The approach, experiments, and results, including the comparison with the BiLSTM-CRF baseline model, are described in section VII. This work is based on a revision of the SmartData corpus, version 1, described in section III. Related work is presented in section II.

II. Related Work

Before the introduction of the transformer architecture [7] and the BERT model [7], state-of-the-art (SOTA) results for the NER task were achieved with BiLSTM-CRF models as desribed in [8]. A thorough investigation of different network design choices and hyperparameters for five common linguistic sequence tagging tasks, including NER, were presented for the English language in [9]. For our work, a BiLSTM-CRF baseline model was generated, including a hyperparameter study that includes a large variety of German word embeddings. This is based on the implementation and the concept of [8], [9]. Transfer learning (TL) for domain-specific NER was studied for the first time in [10], investigating de-identification of electronic health records. A LSTM-network was trained on a large source dataset and retrained with varying training size on a target data set for analyzing the impact of the training data size. Compared to a randomly initialised network, the highest improvement was obtained with the smallest training data size (5%) which decreased with increasing size of training data. A similar study in the biomedical domain [11] assessed the effect of transfer learning on the performance of NER based on LSTM-CRF on 23 different datasets covering four different types of biomedical entity classes. Compared to a SOTA baseline, with transfer learning, an average reduction in error of approximately 11% was observed for their experiments. They also reported that performance gains from transfer learning diminish as the number of training examples used for the target training set increases. [10], [11] motivated Q2 for our application and network architecture. Note, that in both of these works the entities in the target domain remain the same as in the source domain.

In contrast [12] focuses on TL for different entity sets in source and target domain, where the source entity sets are more coarse than the target entity sets. BiLSTM-CRF models, which are pre-trained on the source domain, are trained for new entities and compared with models without pre-training. In this case, it was not possible to identify a clear trend for pre-trained models, except a slight improvement for small target training sets independent from the selected entity subsets. In [13], a target domain is considered, that only consists of new entities. A pre-trained BiLSTM-CRF model is refined on the target data only, which are labeled with the new entities. They showed that the recognition of the considered entity classes can be improved without degrading the recognition of the other entity types. For the experiments Italian language text data from the industrial domain was used. In our work these investigations, presented in [12], [13], motivated Q1. However, our experiments not only cover fine-tuning for each of the 16 entities within the SmartData corpus, but also a comparison with models, which are simultaneously trained for all of the 16 entities.

With the success of the transformer based BERT model [4], NER was treated as a downstream task, using pre-trained models. In [4] already state-of-the-art results were reported for the CoNLL-2003 NER task [14]. Only recently, a German single language model (GottBERT) [15] was published, which is based on the robustly optimised BERT modification, called RoBERTa [16]. This model was evaluated on the two NER tasks CoNLL-2003 and GermEval 2014 [17] on the German portion of the OSCAR data set [18]. These NER tasks, however, only consists of four entities (person, location, organisation, miscellaneous). Based on our knowledge, there is no previous work which considers fine-tuning BERT models for NER on the German SmartData corpus, thus this addresses Q3.

In the work presented here, we first aim at derive new models from previously trained models to recognize completely new entity classes on the German SmartData text corpus. Secondly, we study the impact of the training data size. Finally, we focus on the BERT models, which have been fine-tuned for German domain-specific NER and compared to BiLSTM-CRF baseline models.

III. German Named Entity Corpus

The experiments described in this paper are based on the German SmartData corpus (version1) [3], which is annotated with traffic and industry-related entities and n-ary relations. The work presented here only focuses on NER and the domain-specific entities. The SmartData corpus is the first German dataset that contains more entities than the most commonly used entities person, organization, place, and miscellaneous, which were introduced in the CONLL 2003 dataset [14]. Furthermore, the SmartData corpus is the only one comprising documents of different genres, namely tweets, RSS and news articles, which differ e.g. in length, length of sentences, or type of language. Text data were collected in the period from January 1, 2016 to 31 March 2016, where about 150 mobility and industry-relevant channels were tracked using 300 keywords. Search terms included traffic-related events such as *traffic jams* or *construction*, as well as large motorways and airports by names. In order to obtain industry-relevant results, keywords like *dismissal* or *insolvency* were used. The documents were annotated with sixteen different entity classes, which are explained in Table I. News articles were truncated after the first 1000 characters, in order to reduce the annotation effort. This results in incomplete sentences at the end of some articles, but it is assumed that enough semantic information is retained.

TABLE I: Definition of Entity Types, Annotated in the German SmartData Corpus

Entity / Concept	Description	Examples
Location (LOC)	General locations	Bayern, Zugspitze, Norden
Location-City (LOC-CIT)	Municipalities, e.g. cities, towns, villages	Berlin, Berlin-Buch, Hof
Location-Street (LOC-STR)	Named streets, highways, roads	Hauptstrasse, A1
Location-Route (LOC-ROU)	Named (public) transit routes	U1, ICE 557, Nürnberg – Hof
Location-Stop (LOC-STO)	Public transit stops, e.g. train stations, bus stops	S+U Pankow, Berlin-Buch
Organization (ORG)	General organizations	Greenpeace, Borussia Dortmund
Organization-Company (ORG-COM)	The subset of organizations that are businesses	Siemens AG, BMW
Person (PER)	Persons	Angela Merkel
OrgPosition (POS)	A person's position within an organization	CEO, Vizepräsident
Date (DAT)	Point in time, date	1. September 2017, gestern
Time (TIM)	Time of day	8:30, 5 Uhr früh
Duration (DUR)	Time periods	mehr als eine halbe Stunde
Distance (DIS)	Distances with unit	5 Kilometer
Number (NUM)	Other numeric entities, e.g. money, percentages	3%, 4 Millionen Euro
Disaster-Type (DIS-TYP)	Man-made and natural disaster types	Erdbeben, Überschwemmung
Trigger (TRI)	Trigger terms or phrases for events	Stau, Streik, Entlassungen

A. *Data Preprocessing*

The experiments, described in this work, are based on a revised version of the SmartData corpus, which in the following is referred to as *revised* SmartData corpus. The revision comprises of the deletion of duplicates, a conversion into the IOB2-format (a.k.a. BIO-format), and a more sophisticated split into training, validation, and test data. The IOB2-format was chosen as it was shown to yield best results on NER with BiLSTM-CRF neural networks [9]. It uses three different tags, namely the B-tag (begin), the I-tag (inside), and the O-tag (outside). Non-entities are labeled with the O-tag. For entities, consisting of several words, the first one is labeled with the B-tag, and the following ones with the I-tag. Entities, consisting of a single word, are also marked with the B-tag.

The original SmartData corpus was randomly split into 50% training and 50% test data. Because of duplicates in the originally published data, this 50/50 split differs from an ideally distributed split w.r.t. the distribution of entities and the document genres. This causes a bias within the training and test set, concerning the genre of the document, as well as the entity classes. In the *revised* SmartData corpus the data is split into training, validation, and test data, taking into account the distribution of document types and entity classes within the data. This distribution is depicted in Figure 1, whereas the y-axis shows the number of entities.

The *revised* SmartData corpus is split into 70% training, 10% validation, and 20% test data. Minimization of the deviation from the ideal splitting according to the document genre was reached through stratified random sampling of the documents. The corresponding algorithm is described in [19].

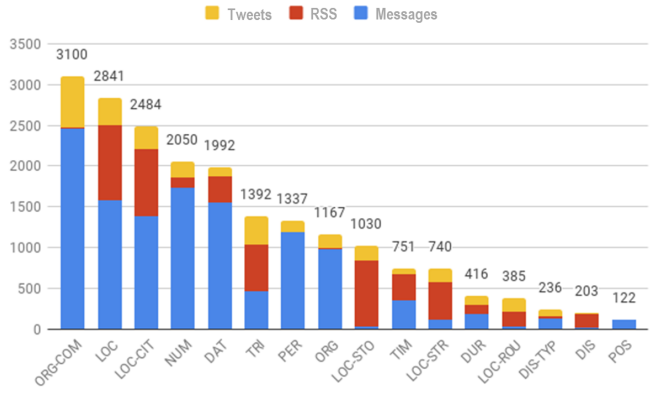


Fig. 1: Entity distribution of the 16 considered classes in the *revised* SmartData corpus.

IV. Named Entity Recognition Approach

The aim of the work, presented in this paper, is to investigate different TL approaches for domain-specific NER. For this purpose, a BiLSTM-CRF baseline model is generated in a first step. In this section, we describe the architecture of a BiLSTM-CRF neural network, followed by the training procedure, we conducted. Finally, we explain the hyperparameter optimization that is performed to obtain a stable baseline model, which is used for the TL experiments, presented in section V. The results of the experiments are presented in VI.

A. *Architecture*

A Long-Short-Term-Memory (LSTM) network [20], as shown in Figure 2, is a recurrent neural network, suitable for processing sequential data. It is equipped with a short-term memory, which is used to reflect dependencies within the sequential input. As depicted in Figure 2, each LSTM-cell receives three input vectors: a memory vector of the previous cell c_{t-1}, the hidden state vector of the previous cell h_{t-1}, and the input vector x_t. The red rectangles display feed-forward neural nets (FFNN) with their respective activation function. These FFNNs are (from left to right): the Forget and Input gate, the candidate layer, and the Output gate. They are responsible for the further processing of the input vectors and their weighting.

The bidirectional extension BiLSTM [21] was introduced for entity recognition. Two different LSTM networks are trained, one of them processes the sequences from left to right, the other one from right to left. The output vectors of the two BiLSTMs are concatenated and passed to a CRF layer, which

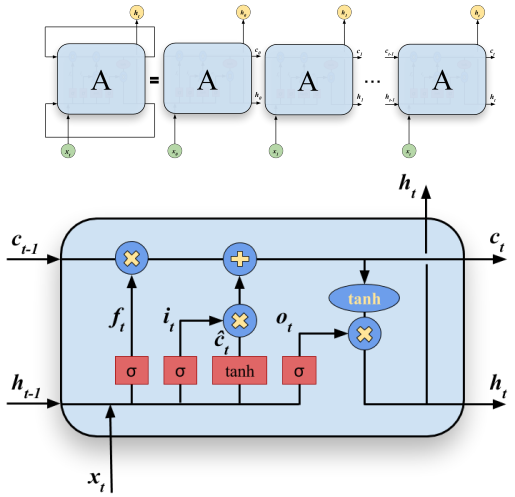


Fig. 2: LSTM-Network - recursive and enrolled representation, detailed presentation of LSTM-cell A

optimizes the recognition of sequences of entity tags. For this purpose, a transition probability matrix is trained for all entity tags. Further improvements are gained by considering not only the words, but also the characters, which are represented by so-called Char-Embeddings. Char-Embeddings are trained either by using an additional BiLSTM-layer [8] or a convolutional neural net (CNN) [22].

B. Training of BiLSTM-CRF Networks

The BiLSTM-CRF models are trained and tested on the *revised* SmartData corpus after the data was split into training, validation, and test data as described in section III. The evaluation of the trained models is based on the *Micro-Average-F1-Score*, which reflects the distribution of the different entity classes in the corpus. According to [14], the F1-Score is calculated for each entity class and weighted by the number of entities within this class. Each model is trained for a maximum of 10 epochs. After each epoch, the *F1-Score* is calculated on the validation set. If this yields the highest score, then the *F1-Score* is calculated on the test data. Early stopping on the validation set is set to 10 epochs. For comparison, the *Macro-Average-F1-Score* is also reported for some experiments, which refers to the sum of unweighted F1-Scores for each entity class divided by the number of entity classes.

C. Hyperparameter Optimization

In order to generate a stable baseline model for the TL experiments, an optimal hyperparameter configuration is aimed to be defined. For English text corpora, a thorough hyperparameter study for five different sequence labeling tasks, including NER, is described in [9]. The German dataset used provides considerably more entities than the English dataset, therefore a new hyperparameter study is conducted based on [9]. The hyperparameter space is depicted in Table II. s_{min} refers to a threshold, defining the minimal number of occurrences of a single normalised word in the training data.

TABLE II: The Hyperparameter Space for Optimization

Hyperparameter	Values	Best
Word-embeddings	FastText [27], Word2Vec [17], [28]	SB Wiki+ Vectors NLP+ SB Tweets
Threshold s_{min}	1,3,5,10	5
POS-tags	yes, no	no
Chunking-Tags	yes, no	no
Char-Embeddings	LSTM, CNN	LSTM
BiLSTM-Layer	1,2,3	1
Units of LSTM-Net	100, 200, 250	200
Dropout	0.1, 0.3, 0.5	0.5
Variational Dropout	0.1, 0.3, 0.5	0.5
Batchsize	8, 16, 32	16
Optimizer	Adam, Nadam, RMSProp	Adam
F1-Score		0.7834

Normalised words are words that are transformed in a rule-based manner, e.g. replacing numbers with specific tokens. If a word is not represented within the word-embeddings and does occur more than s_{min} times within in the training data, an embedding is sampled from the zero-mean normal distribution with a standard deviation of 0.25. CNN stands for Convolutional Neural Network, where Adam, Nadam, RMSProp refer to the optimizers described in [23]–[25], respectively. Combinations of domain-specific and unspecific word-embeddings may improve the results on domain-specific texts [26]. In the context of this hyperparameter study, therefore, not only different word embeddings but also different combinations of embeddings are considered.

The optimization process is organized as follows. First, randomized testing yields a set of ten configurations considered to be the best. Given these configurations, ten different word-embeddings and combinations therefrom are tested again. Finally, the ten best configurations of the randomized testing, each of them with one of the two best word-embeddings fixed, are used to test the remaining hyperparameters independently. The selection of the best hyperparameters are selected w.r.t. the averaged F1-Score on the validation set. For this reason, the effects of the hyperparameters are not only examined on the F1-Score of the test set, but also on the F1-Score of the validation set. In total, 2765 models are trained and 1348 different configurations are tested within the randomized testing. This yields a preselection for the further hyperparameter study, where another 5.600 models are trained.

In Table II the best hyperparameter configuration according to the F1-Score on the validation set and the corresponding F1-Score are depicted in the rightmost column. The word embedding *SB Wiki+Vectors NLP+SB Tweets* refers to the concatenation of the word embeddings *SB Wiki*, *VectorsNLP*, and *SB Tweets*, yielding word-embeddings of dimension 500. *SB Wiki*, and *SB Tweets* both are trained with the Fasttext algorithm [29] on two million German Wikipedia articles and 50 million German Tweets, respectively. *VectorsNLP* are trained with the "Word2Vec Continuous Skipgram" algorithm on the German CoNLL17 data set [30] and are available online [31].

The hyperparameter configuration, obtained by the optimization process described above, is used for the generation

of the baseline model and the TL experiments, described in section V. More details about this hyperparameter study can be found in [32].

V. Transfer Learning with BiLSTM-CRFs

In this section, we present TL experiments based on BiLSTM-CRF networks. The task under consideration is NER on the *revised* SmartData corpus for each of the 16 annotated entity classes. The investigation comprises the impact of a TL approach on this task and the dependency of this approach on the training data size. The TL experiments, conducted in this work, are homogeneous, i.e. the feature space of the source domain is the same as of the target domain, referring to the *revised* SmartData corpus.

A. Experimental Setup for each Entity Class

The *revised* SmartData corpus consists of documents, which are annotated with 16 different entity classes $E_1, ..., E_{16}$ (see Table I). For each of these entity classes E_i a new dataset is derived from the *revised* SmartData corpus as follows. Each entity annotation of class E_i is removed, whereas all the other entity annotations for the remaining fifteen classes are kept. The derived dataset will be referred to as $D(!E_i)_{base}$. Based on $D(!E_i)_{base}$, a BiLSTM-CRF network, called M_{source}, is trained for the recognition of the remaining fifteen entities. It acts as a base model for the TL experiment for class E_i. The training is run ten times.

For conducting the TL experiments, a target data set $D(E_i)_{tar}$ is created by removing all entity annotations but the entity annotations of the considered entity class E_i. The respective text sequences within training, validation and test dataset are instead replaced by the non-entity tag 'O'. Based on the data set $D(E_i)_{tar}$, three different BiLSTM-CRF nets are trained ten times each for the recognition of the entity class E_i. The three different BiLSTM-CRF nets are named $M(rand)_{E_i}$, $M(all)_{E_i}$, and $M(noLL)_{E_i}$, and are initialised as follows:

1) $M(rand)_{E_i}$: a randomly initialised BiLSTM-CRF neural net serving as a baseline model
2) $M(all)_{E_i}$: all weights of M_{source} are transferred into $M(all)_{E_i}$
3) $M(noLL)_{E_i}$: all weights of M_{source} without the last BiLSTM layer are transferred, and a new output layer with size three is added into $M(noLL)_{E_i}$. The output layer corresponds to the three tags (O, B, I), labeling non-entities, single word entities, and multiple word entities as described in section III.

B. Influence of Training Data Size

The training data of $D(E_i)_{tar}$ is split into seven parts, each of them containing 10% of the data set $D(E_i)_{tar}$. This split is stratified according to the number of entity occurrences within one sentence. Combining these smaller data sets yields seven training data sets sized by 10%, 20%, 30%, 40%, 50%, 60%, and 70% of $D(E_i)_{tar}$, respectively. The TL experiments as described above are run for each size of training data. The optimized hyperparameter configuration, as described in section IV, is used for all of the experiments. As for the baseline, the training is run ten times on each data set in order to reduce statistical noise. Validation is based on the averaged F1-Score on the validation data. The experiments as described above are run for each of the 16 entity classes. Thus, the TL experiments comprises the training of 220 neural networks for each entity class, i.e. 3520 models in total.

For comparison, models that recognize all 16 entity classes at the same time are considered as well.

VI. Results Transfer Learning Experiments

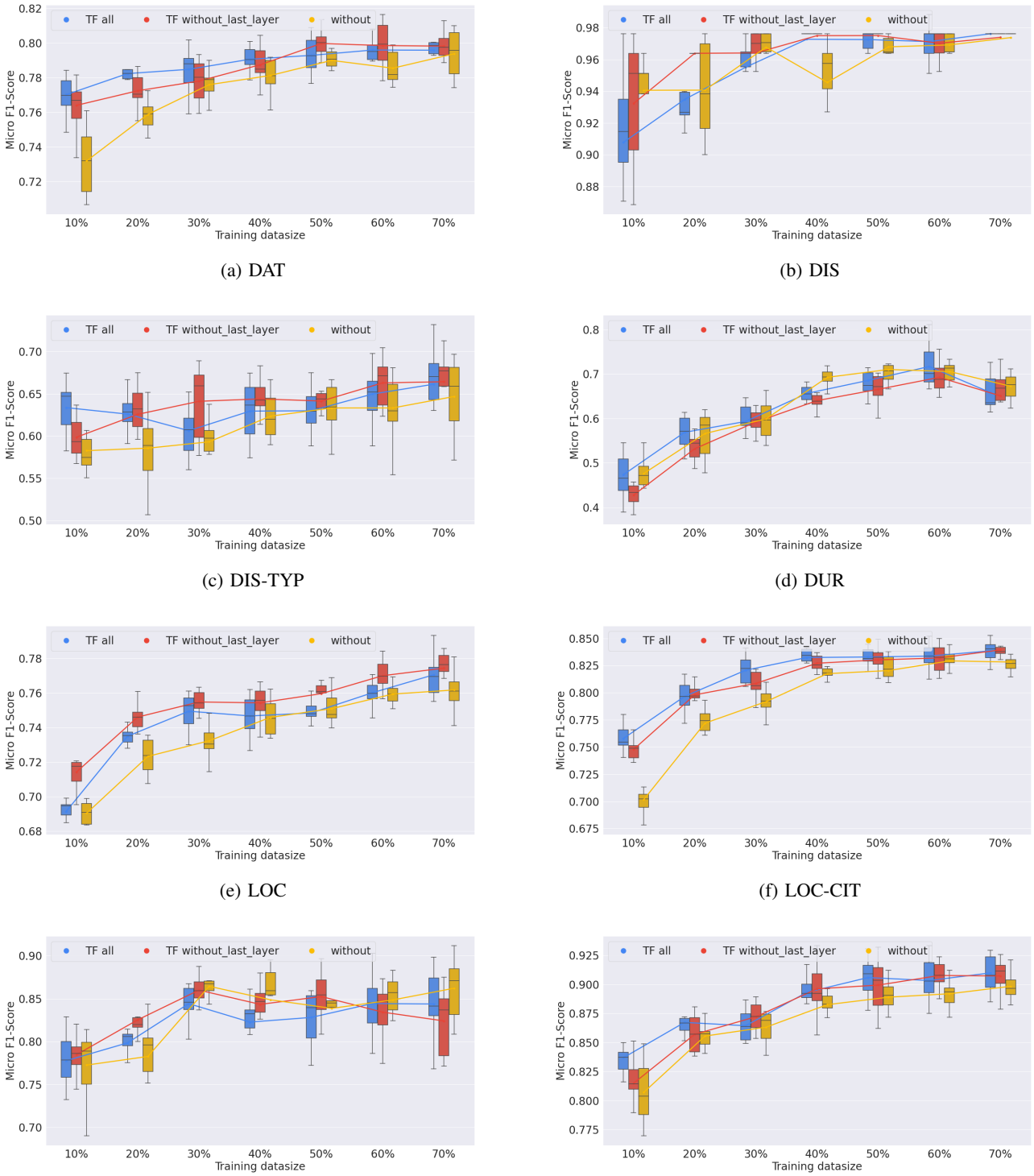
The results of the TL experiments show that, in principle, the knowledge about the other fifteen entity classes on average helps in recognizing a new entity type. The influence decreases with increasing training data size. This holds for most of the considered entity classes, which we investigated in our studies. To illustrate this, in Table III the averaged F1 score over all available sixteen entity classes is given for each trained model. The individual results for all the 16 independent entity classes are shown in Figure 3 and 4. We use box-plot methods, presenting the min and max of the averaged F1 score as well as the 25% and the 75% percentiles. The lines between the boxes connect the mean values in order to underline average behaviour. The yellow lines refer to the randomly initialised baseline model $M(rand)_{E_i}$, the blue lines refer to the model $M(all)_{E_i}$, receiving all weights from the source model M_{source}, and the red lines refer to the model $M(noLL)_{E_i}$, receiving all weights, except for the last BiLSTM layer from the source model M_{source} before being trained further.

We observe, that the TL model, which takes over all weights to the new network, performs slightly better than the model where the last layer is left out. Although the differences are small, especially for larger training data sizes, the general behavior is consistent over almost all of the compared dataset sizes and entities. This behavior appears counter intuitive, since the model first needs to forget about the previously learned 15 entity classes. However, adapting a model to one of 15 classes might be less difficult than reducing the classes from 15 to 1. This assumption could be verified by running further experiments, generating base models M_{source} (see section V) with different, smaller numbers of entities. The differences between the considered models in our experiments are largest with the smallest training dataset sizes.

We further observe F1-Scores, which are in general higher for all TL models compared to the baseline. This might be due to the source models M_{source} that were trained on the entire training data set and thus can probably handle the embedding vector representation of the individual words better than a neural net trained only on a fraction of the data set. Only the entity class 'POS' does not follow the general trend. This is most likely due to the fact, that for this class only 122 messages are included in the full data set, and our splitting method does not guarantee to have sufficient training data in each data sample. This can also be seen in the drastically lower F1-Score for all models for the entity class 'POS' (see chart

transfer learning model $i = \{1, ..., 16\}$	averaged F1 score for amount of training data (% of full data set) 10	20	30	40	50	60	70
without: baseline $M(rand)_{E_i}$	0.672	0.715	0.737	0.752	0.763	0.769	0.771
TF_all: all weights $M(all)_{E_i}$	**0.698**	**0.734**	**0.748**	0.758	**0.766**	**0.773**	**0.774**
TF_without_last_layer: $M(noLL)_{E_i}$	0.682	0.733	0.744	**0.759**	0.765	0.770	0.773

TABLE III: Averaged F1 scores over all entity classes for different training data sizes.



(a) DAT

(b) DIS

(c) DIS-TYP

(d) DUR

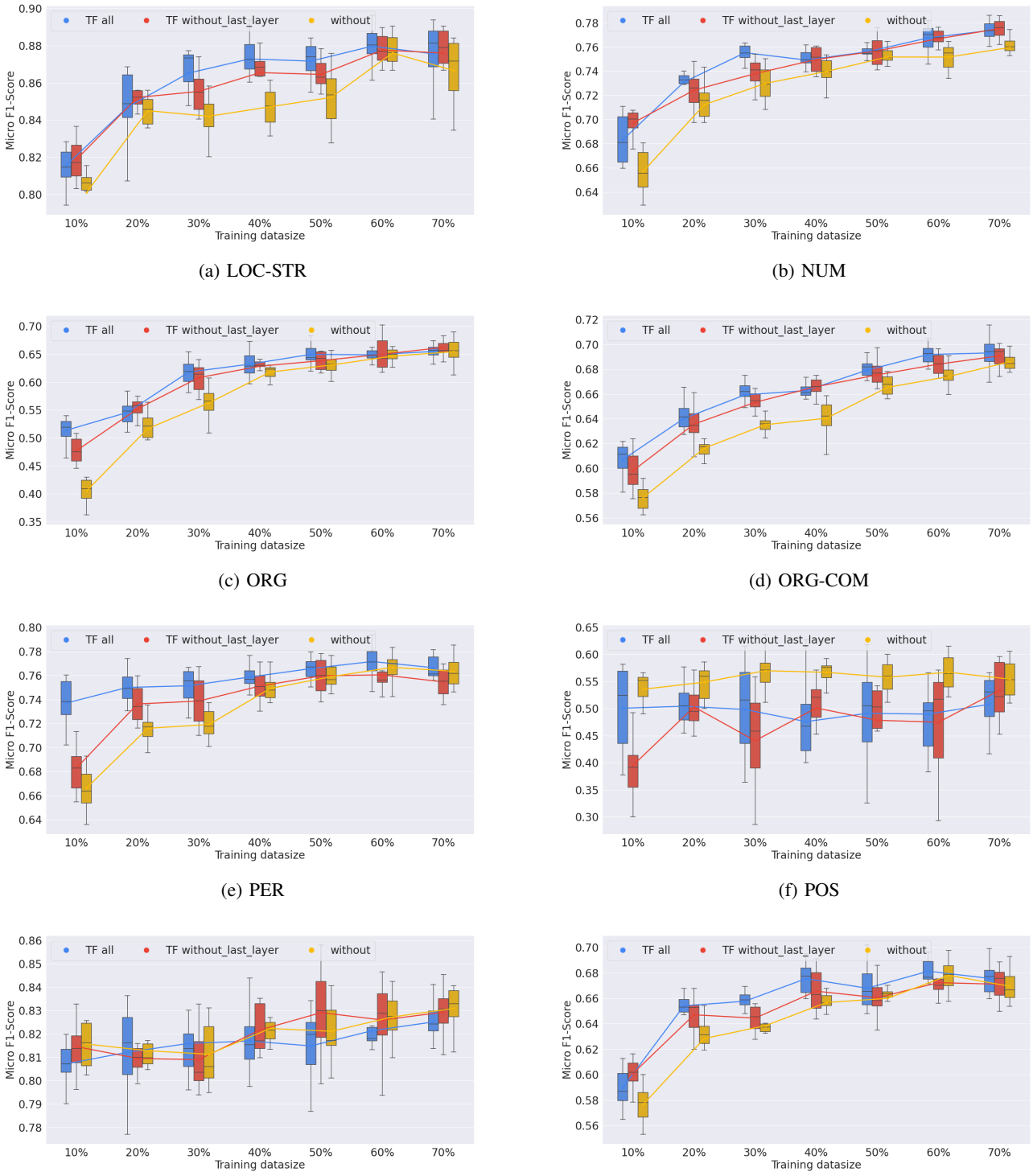
(e) LOC

(f) LOC-CIT

(g) LOC-ROU

(h) LOC-STO

Fig. 3: TL results for varying training data size for entity 1-8.



(a) LOC-STR

(b) NUM

(c) ORG

(d) ORG-COM

(e) PER

(f) POS

(g) TIM

(h) TRI

Fig. 4: TL results for varying training data size for entity 9-16.

(f) in Figure 4). Experiments without the outlier class 'POS' show comparable results concerning the TL model with all weights compared to the model without the last layer.

For each entity class, we choose those models of the TL experiments that yield the highest average F1-Score on the validation set, trained on training data of size 70%. For each entity class, we compare the average F1-Score on the validation set and the test set with the average F1-Scores of the multi-entity models MULT, trained for all entities at the same time. Table IV shows that, *on average*, the multi-entity models perform better on the test data. However, there were single TL models that performed better on some entity

TABLE IV: Comparison of MULT model with best models from TL experiments.

Entity	MULT Validation	MULT Test	Best Models from TL-Experiments Validation	Best Models from TL-Experiments Test
DAT	0.7790	**0.8015**	**0.7883**	0.7983
DIS	0.9885	0.9716	**0.9977**	**0.9762**
DIS-TYP	0.6261	**0.6989**	**0.6953**	0.6613
DUR	0.7087	**0.7177**	**0.7379**	0.6584
LOC	0.7880	**0.7803**	**0.7951**	0.7706
LOC-CIT	**0.8365**	**0.8445**	0.8209	0.8394
LOC-ROU	0.7997	0.8419	**0.8523**	**0.8437**
LOC-STO	0.8512	**0.9123**	**0.8941**	0.9073
LOC-STR	0.8858	0.8687	**0.9051**	**0.8756**
NUM	**0.7918**	**0.7811**	0.7870	0.7736
ORG	0.5744	**0.6830**	**0.6396**	0.6545
ORG-COM	0.7516	**0.7024**	**0.7539**	0.6931
PER	0.8444	0.7567	**0.8695**	**0.7665**
POS	0.5653	**0.5510**	**0.5982**	0.5080
TIM	0.8451	0.8186	**0.8733**	**0.8309**
TRI	0.7333	0.6737	**0.7390**	**0.6757**
Total (Micro)	0.7834	**0.7756**	**0.7948**	0.7679
Total (Macro)	0.7731	**0.7752**	**0.7967**	0.7646

classes. This concerns the entity classes TIM (Time), TRI (Trigger), PER (Person), LOC-STR (Location-Street), LOC-ROU (Location-Route), and DIS (Distance). In case of the entity class TIM (Time), one reason could be an entity length of 1 or 2 in $\sim 95\%$ of the annotated sequences. Furthermore, the description of time in most of the cases follows simple rules.

VII. Transfer Learning Using BERT Models

In contrast to the transfer learning approach, reported in previous parts of this paper, the concept and application of the Bidirectional Encoder Representations for Transformers (BERT) [4] allow a transfer of knowledge in a different way. A deep neural network is pre-trained on two NLP tasks in an unsupervised manner and will then be fine-tuned on special NLP tasks (downstream tasks). The architecture and the methodological concepts of this approach are described in detail in [4], [7]. It has had significant impact on the state-of-the-art performance of many different NLP tasks and increased the relevance of pre-trained models. Since then, this approach has been applied and improved for many NLP applications and different languages (see e.g. [15], [33]). The first pre-trained models, that have been released by Google and others, were trained on general domain corpora such as news articles and Wikipedia text documents. However, in order to improve NLP tasks like NER in specific domains, pre-training should be conducted on large domain-specific corpora. This is already done for some domain-specific Bert-based models. The BioBert model [34] was trained on text data of the biological and medical domain and showed better results than the English BERT model, released by Google, when being applied to domain-specific NER. Another pre-trained domain-specific BERT model is SciBert [35], which has been trained on text data of biology, medicine, and computer science. In our work, however, due to the small size of the SmartData corpus and the lack of large-size domain-specific German training corpora, we focus on BERT models, which are pre-trained on general domain corpora, and refine them on the NER task on the German SmartData corpus. The models under consideration are the multi-lingual BERT model, provided by Google [6], and the German BERT model, released by Deepset.AI [5]. This model was the first German model, which has been released and at that time the only available one. In the following paragraphs, the refinement of the pre-trained models is described, followed by a parameter optimization study, and the comparison with the BiLSTM-CRF baseline model as described in section IV.

TABLE V: Hyperparameter Space for Refinement Training.

Hyperparameter	Values	Best
BERT model	multi-lingual, German	German
Learning rate (Adam)	1e-5, 2e-5, 3e-5, 5e-5	1e-5
Batchsize	8, 16, 32	8

A. Refining Pre-Trained BERT Models

In our work, we refine pre-trained BERT models w.r.t. NER on the German SmartData corpus. The BERT models under consideration are the multi-lingual BERT model [6], and the German BERT model [5]. In the following, the process of refinement is described, which also comprises pre-processing the SmartData corpus. The BERT architecture requires sub-words instead of words as input sequence. Therefore, using the tokenizer of the BERT model, each word in the corpus is separated into word pieces and the first word piece is indicated by the special token "##". Unknown tokens are marked with the "Unknown"-token "[UNK]". Entity tags are assigned to initial word pieces iteratively. The maximal sequence length is set to 128, not counting the "[CLS]" and "[SEP]" tokens, which are used to mark the beginning and the end of a sequence. We use padding for short sequences, whereas longer sequences were cut off.

For refinement training of the given BERT models, the output of the last layer is fed into a feed-forward network. According to [4], the loss is calculated on the last layer of the network using the optimizer, Adam [23]. We use the PyTorch-Transformers Framework, provided by Huggingface [36], and the framework FARM [37].

B. Parameter Optimization

The refinement training includes an investigation of the hyperparameter space as depicted in Table V. For each configuration 10 models are trained, resulting in 240 trained models. Early stopping after five epochs is used during the training process.

The multi-lingual and the German BERT model are both BERT-base-models [4], i.e. each of them consists of 12 layers with a hidden size of 768 and each model consists of 12 attention heads. Both models are "cased" models, differentiating between lower and upper cases. The multi-lingual model was pre-trained on the Wikipedia dumps of the most common 104 languages. This results in a small fraction of German training data. For this reason, the vocabulary of the tokenizer of the

TABLE VI: F1-Scores Depending on the BERT-Model

BERT-Model	Validation Micro-F1 Min	Max	Avg	Std Dev	Med	Test Micro-F1 Min	Max	Avg	Std Dev	Med
German	0.7555	**0.7870**	**0.7733**	0.0056	**0.7731**	**0.7530**	**0.7857**	**0.7699**	0.0058	**0.7696**
multi-lingual	**0.7597**	0.7816	0.7713	0.0049	0.7718	0.7471	0.7806	0.7626	0.0061	0.7629

Fig. 5: F1-Scores depending on the BERT model

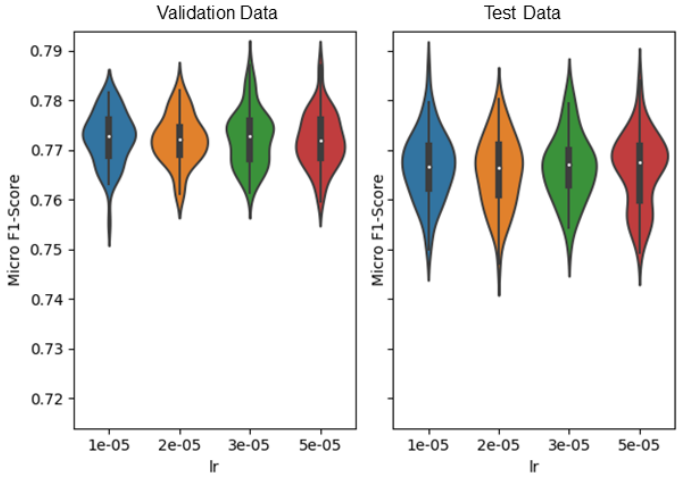


Fig. 6: F1-Scores depending on learning rates

multi-lingual model, which was trained with WordPiece [38], is very strongly influenced by different languages and thus has a limited number of subwords that are appropriate for the German language. For the German model the tokenizer was trained using SentencePiece [39]. Therefore we investigated the coverage of the data set by the subwords provided by the tokenizers. It turns out, that the multi-lingual model needs 236.704 subwords (1,56 subwords per word) in order to cover the entire SmartData corpus whereas the German model only needs 209.925 subwords (1,38 subwords per word). Thus we expected a better performance of the German model, which is confirmed by the experiments, where the German models yield better F1-scores (see Figure 5). Hereby, the difference on the test set is higher than on the validation set (see Table VI). Compared to the German model, the multi-lingual model needs, on average, two more epochs for termination.

We further investigated, whether certain entity types are better recognised by one model than by the other. Although in general the German model achieves the better F1 scores, the multi-lingual model achieves the better F1 scores in recognising durations (average difference 2.8-4.8%) and positions within companies (average difference 5.2-5.5%). The German model, on the other hand, is better in recognising organisations (average difference 2.3-2.6%) and disaster types (average difference 4.1-9.2%). For the detection of people and roads, the two models achieves comparable results on the validation set. However, on the test set, the German model achieves F1 scores that are 2.33% and 2.35% higher on average, respectively. For the other entity classes, either the differences between the models are smaller or they are less clear. For the recognition of time e.g., on average the multi-lingual model achieves better F1 scores on the validation set whereas the German model achieves higher F1 scores on the test set.

1) Learning rate: The investigation of different learning rates did not show any clear result (see Figure 6). On average, the best results were yielded with a learning rate of 1e-5. However, the differences compared with other learning rates were very small.

2) Batch Size: The experiments with smaller batch sizes show better results. A batch size of 8 yielded the best results (see Table VII). This might be due to the small size of the SmartData corpus.

In Table VI and VII, the standard deviation is presented. The values vary between 0.49% (Table VI: multi-lingual model, validation set, Table VII: batch size 32, validation set) and 0.75% (Table VII: batch size 8, test set). This indicates statistically stable models, if we consider the recognition of all entities. If we consider each single entity, this might be different as the number of annotated entities varies within the data. Looking at the box-plots in Figure 3 and 4, we observe a remarkable variance. This concerns the averaged F1 scores, which vary e.g. between below 0.70 for the entities ORG, POS, TRI in Figure 4, chart (c), (f), and (h), and up to more than 0.80 for the entities DAT, DIS, LOC-CIT, LOC-ROU, LOC-STO, LOC-STR, and TIM (see Figure 3, chart (a), (b), (f), (g), and (h), and Figure 4, chart (a) and (g)). More insight could give the calculation of p-values. However, this needs to make the assumption of independence of the models, which is not obvious in the case of our TL experiments. The varying distribution of entities within the data also may distort the results. Thus, the box-plots for the single entities seems to be more appropriate for the analysis of our results.

TABLE VII: F1-Scores Depending on Batch Sizes

Batch size	Validation Micro-F1 Min	Max	Avg	Std Dev	Med	Test Micro-F1 Min	Max	Avg	Std Dev	Med
8	**0.7616**	**0.7870**	**0.7752**	0.0052	**0.7749**	0.7471	**0.7857**	**0.7684**	0.0075	**0.7685**
16	0.7614	0.7813	0.7719	0.0045	0.7717	**0.7524**	0.7840	0.7660	0.0065	0.7665
32	0.7555	0.7835	0.7698	0.0049	0.7698	0.7494	0.7787	0.7644	0.0066	0.7646

TABLE VIII: F1-Scores Depending on Entities

Entity	Validation F1 BiLSTM Max	BiLSTM Avg	BERT Max	BERT Avg	Test F1 BiLSTM Max	BiLSTM Avg	BERT Max	BERT Avg
DAT	0.8075	0.7664	**0.8122**	**0.7755**	**0.8290**	**0.8028**	0.8243	0.7875
DIS	**1.0000**	**0.9822**	**1.0000**	0.9601	**0.9882**	**0.9715**	0.9762	0.9551
DIS-TYP	0.7556	**0.6203**	**0.7826**	0.5776	**0.7816**	**0.6813**	0.7470	0.6247
DUR	**0.8049**	**0.7146**	0.7778	0.6652	**0.8295**	**0.7222**	0.6780	0.5995
LOC	**0.8318**	**0.7896**	0.8037	0.7617	**0.8071**	**0.7736**	0.7885	0.7575
LOC-CIT	0.8646	**0.8323**	**0.8671**	0.8232	**0.8719**	**0.8439**	0.8525	0.8160
LOC-ROU	**0.9189**	0.7950	0.8767	**0.8119**	**0.9067**	**0.8273**	0.8794	0.8146
LOC-STO	0.9038	0.8519	**0.9307**	**0.8832**	**0.9565**	**0.9096**	0.9364	0.8935
LOC-STR	0.9306	0.8826	**0.9437**	**0.8972**	0.9084	0.8603	**0.9296**	**0.8844**
NUM	**0.8249**	**0.7836**	0.7929	0.7477	**0.8234**	**0.7891**	0.8132	0.7792
ORG	0.6754	0.5700	**0.6983**	**0.6269**	0.7342	**0.6741**	**0.7403**	0.6738
ORG-COM	0.7778	0.7370	**0.7780**	**0.7396**	0.7349	0.6968	**0.7433**	**0.7084**
PER	0.8990	0.8373	**0.9039**	**0.8518**	0.8127	0.7708	**0.8436**	**0.8099**
POS	**0.8148**	0.5490	0.7692	**0.5552**	**0.7170**	**0.5386**	0.6809	0.5217
TIM	0.8944	**0.8358**	**0.8970**	0.8202	0.8598	**0.8219**	**0.8599**	0.8017
TRI	**0.7754**	**0.7197**	0.7463	0.6974	0.7236	0.6758	**0.7308**	**0.6848**
Total	**0.7921**	**0.7762**	0.7870	0.7723	**0.7918**	**0.7740**	0.7857	0.7663

C. Comparison with Baseline Model BiLSTM-CRF

The investigation of pre-trained BERT models and their refinement for domain-specific NER is concluded with a comparison with the BiLSTM-CRF baseline model, trained as described above (see section IV). This comparison was conducted on entity level. Table VIII shows the maximum and the average F1-scores. On average, the BiLSTM architecture scored 0.24 % higher than the BERT architecture on the validation set and 0.77% higher on the test set. However, the number of trained BiLSTM models is more than ten times higher than the number of trained BERT models. Therefore, since the BERT model yields a higher average value for a single entity, this can be seen as a clearly better recognition of this entity. This can be observed for the recognition of persons (PER), commercial organizations (ORG-COM), and streets (LOC-STR), which are all proper names. On the other hand, city names (LOC-CIT) were better recognized by BiLSTM models.

VIII. Conclusion

In this paper, we presented TL approaches for entity recognition on a German text corpus. The motivation of these investigations was to overcome the problem of lacking domain-specific training data by transferring structural information from one model to another. Our investigations addressed the following questions:

- Q1 Is it possible to transfer information within a German text corpus from one specific domain to another specific domain?
- Q2 What is the influence of the training data size on this special case?
- Q3 Does domain-specific NER also profit from general domain knowledge and general task knowledge as provided by BERT language models?

Our experiments are based on the *revised* SmartData corpus, i.e. the pre-processed SmartData corpus enriched with BIO-annotations and a revised splitting into training, validation, and test data.

We considered two different approaches. First, our investigation focus was on the transfer from a specific domain to a different specific domain, i.e. from a model M_{source} trained for the recognition of entity classes $E_1, ..., E_n$ within domain Dom_{source} to a model M_{target} intended to be used for the recognition of a *new* entity class, E_{new} within a different specific domain Dom_{target}. In particular, the influence of the size of the target training data set was studied.

Due to the lack of further domain-specific German text corpora for entity recognition, we defined the source and target feature space to be the same, differing only with respect to annotations. For our experiments we used BiLSTM-CRF neural networks.

Our investigations showed that the transfer of structural information, gained beforehand by training entity recognition for some classes $E_1, ..., E_n$, improves the entity recognition for a new entity class, E_{new}. The SmartData corpus provides entity annotations in different domains. Thus, it is possible to transfer information from one specific domain to another specific domain (Q1). However, in our TL experiments, the entities of the source domain belong to both domains, namely traffic and industry, whereas the entity in the target domain only belongs to one of these domains. Further investigations need a more strict separation between source and target

domain, in order to obtain a clear picture.

Improvements are highest when the size of the target training data is small, and depend on the entity class under consideration (Q2). Further experiments showed that learning all entity classes in common yields even better results.

Future work should be based on text corpora from different domains such that source and target domain not only differ in annotations but also in occurrence in the vocabulary. This would enable a study of the described TL approaches in combination with transfer of domain-specific knowledge as described in [40], where a pre-trained model is combined with a model trained on the new data. This is done by solving a word vector alignment problem and thus adapting word vector models and text classifiers to new data.

A comparison with other approaches could provide some indication of the quality of the results. This may include different learning approaches but also different text corpora.

In the second part of our study we addressed the fine-tuning of BERT models on the downstream task NER. In this case, transfer learning refers to the usage of a pre-trained model, which was trained on a large general domain text corpus and refined for the domain-specific downstream task NER. We fine-tuned a multi-lingual and a German BERT model for German NER on the *revised* SmartData corpus and conducted a hyperparameter study. Experiments showed better results for the German BERT model, however for some entities the multi-lingual model performed better. A comparison with the generated BiLSTM-CRF baseline model showed that on average, BERT models performed almost as well as the BiLSTM-CRF model. This is remarkable as the number of experiments with the BiLSTM-CRF baseline model is more than ten times higher than with the BERT models.

Further investigations may include the transfer of word-embeddings as studied in [40]. Due to the lack of training data in many domains, a systematic study of the required size of training data could be helpful. This could offer guidelines for the extent to which the costly provision of domain-specific training corpora is strongly required in order to reach acceptable performance results on NLP tasks in low-resource domains, or for corpora of different languages.

As this work was motivated by real word applications, we conclude that transfer learning is very helpful in low-resource domains. However, there remains a gap due to the lack of domain-specific knowledge, which needs to be provided by larger domain-specific training corpora or other approaches as proposed e.g. in [41]. The *revised* SmartData corpus, including all results of this work will be made publicly available, and can also be provided upon request.

ACKNOWLEDGMENT

We would like to thank our colleagues Christoph Lehmann and Neringa Jurenaite for interesting discussions, and the reviewers for their detailed and helpful comments. This work was supported by the German Federal Ministry of Education and Research (BMBF, 01IS14014A-D) by funding the competence center for Big Data and AI "ScaDS.AI Dresden/Leipzig". The authors gratefully acknowledge the GWK support for funding this project by providing computing time through the Center for Information Services and HPC (ZIH) at TU Dresden on the HRSK-II.

REFERENCES

[1] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021.

[2] K. Weiss, T. Khoshgoftaar, and D. Wang, "A survey of transfer learning," vol. 9, no. 3, 2016. [Online]. Available: https://doi.org/10.1186/s40537-016-0043-6

[3] M. Schiersch, V. Mironova, M. Schmitt, P. Thomas, A. Gabryszak, and L. Hennig, "A german corpus for fine-grained named entity recognition and relation extraction of traffic and industry events," in *Proceedings of the 11th International Conference on Language Resources and Evaluation*. European Language Resources Association, 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[5] Deepset.AI. (2019, last visited 12th of January, 2022) German bert model. [Online]. Available: https://deepset.ai/german-bert

[6] Google. (2018, last visited 12th of January, 2022) Multi-lingual bert model. [Online]. Available: https://storage.googleapis.com/bert_models/2018_11_03/multilingual_L-12_H-768_A-12.zip

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[8] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016. [Online]. Available: https://www.aclweb.org/anthology/N16-1030

[9] N. Reimers and I. Gurevych, "Optimal hyperparameters for deep lstm-networks for sequence labeling tasks," *CoRR*, vol. abs/1707.06799, 2017. [Online]. Available: https://arxiv.org/pdf/1707.06799.pdf

[10] J. Y. Lee, F. Dernoncourt, and P. Szolovits, "Transfer learning for named-entity recognition with neural networks," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. [Online]. Available: https://www.aclweb.org/anthology/L18-1708

[11] J. M. Giorgi and G. D. Bader, "Transfer learning for biomedical named entity recognition with neural networks," *Bioinformatics*, vol. 34, no. 23, pp. 4087–4094, 06 2018. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty449

[12] J. D. Rodriguez, A. Caldwell, and A. Liu, "Transfer learning for entity recognition of novel classes," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1974–1985. [Online]. Available: https://www.aclweb.org/anthology/C18-1168

[13] L. Chen, A. Moschitti, G. Castellucci, A. Favalli, and R. Romagnoli, "Transfer learning for industrial applications of named entity recognition," in *NL4AI@AI*IA*, 2018.

[14] E. F. Sang and F. De Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," *arXiv preprint cs/0306050*, 2003.

[15] R. Scheible, F. Thomczyk, P. Tippmann, V. Jaravine, and M. Boeker, "Gottbert: a pure german language model," *CoRR*, vol. abs/2012.02110, 2020. [Online]. Available: https://arxiv.org/abs/2012.02110

[16] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[17] N. Reimers, J. Eckle-Kohler, C. Schnober, J. Kim, and I. Gurevych, "Germeval-2014: Nested named entity recognition with neural networks," 2014. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:gbv:hil2-opus-3023

[18] P. J. Ortiz Suárez, B. Sagot, and L. Romary, "Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures," in *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*, P. Bański, A. Barbaresi, H. Biber, E. Breiteneder, S. Clematide, M. Kupietz, H. Lüngen, and C. Iliadi, Eds. Cardiff,

United Kingdom: Leibniz-Institut für Deutsche Sprache, Jul. 2019. [Online]. Available: https://hal.inria.fr/hal-02148693

[19] S. Torge, W. Hahn, R. Jäkel, and W. E. Nagel, "Corpus and baseline model for domain-specific entity recognition in german," in *2020 6th IEEE Congress on Information Science and Technology (CiSt)*, June 2020, pp. 314–320.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[22] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf." in *ACL (1)*. The Association for Computer Linguistics, 2016. [Online]. Available: http://dblp.uni-trier.de/db/conf/acl/acl2016-1.html#MaH16

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[24] T. Dozat, "Incorporating nesterov momentum into adam," 2016.

[25] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited on*, vol. 14, p. 8, 2012.

[26] M. Joshi, E. Hart, M. Vogel, and J.-D. Ruvini, "Distributed word representations improve ner for e-commerce," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 2015, pp. 160–167.

[27] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1638–1649. [Online]. Available: https://www.aclweb.org/anthology/C18-1139

[28] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. USA: Curran Associates Inc., 2013, pp. 3111–3119. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999792.2999959

[29] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[30] M. Fares, A. Kutuzov, S. Oepen, and E. Velldal, "Word vectors, reuse, and replicability: Towards a community repository of large-text resources," in *Proceedings of the 21st Nordic Conference on Computational Linguistics*, 2017.

[31] U. O. Language Techology Group. Nlpl word embeddings repository. University Oslo. [Online]. Available: http://vectors.nlpl.eu/repository/

[32] S. Torge, W. Hahn, and R. Jäkel, "Domain-specific entity recognition in german: Extended corpus and baseline model," submitted, 2020.

[33] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how BERT works," *Trans. Assoc. Comput. Linguistics*, vol. 8, pp. 842–866, 2020. [Online]. Available: https://transacl.org/ojs/index.php/tacl/article/view/2257

[34] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "Biobert: a pre-trained biomedical language representation model for biomedical text mining," *Bioinform.*, vol. 36, no. 4, pp. 1234–1240, 2020. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz682

[35] I. Beltagy, K. Lo, and A. Cohan, "Scibert: A pretrained language model for scientific text," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Association for Computational Linguistics, 2019, pp. 3613–3618. [Online]. Available: https://doi.org/10.18653/v1/D19-1371

[36] HuggingFace. (last visited 12th of January, 2022) Huggingface transformers library. [Online]. Available: https://github.com/huggingface/transformers

[37] Deepset.AI. (last visited 12th of January, 2022) Farm framework. [Online]. Available: https://github.com/deepset-ai/FARM

[38] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[39] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71. [Online]. Available: https://www.aclweb.org/anthology/D18-2012

[40] P. Bojanowski, O. Celebi, T. Mikolov, E. Grave, and A. Joulin, "Updating pre-trained word vectors and text classifiers using monolingual alignment," *CoRR*, vol. abs/1910.06241, 2019. [Online]. Available: http://arxiv.org/abs/1910.06241

[41] T. Schick and H. Schütze, "Exploiting cloze-questions for few-shot text classification and natural language inference," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, Apr. 2021, pp. 255–269. [Online]. Available: https://www.aclweb.org/anthology/2021.eacl-main.20