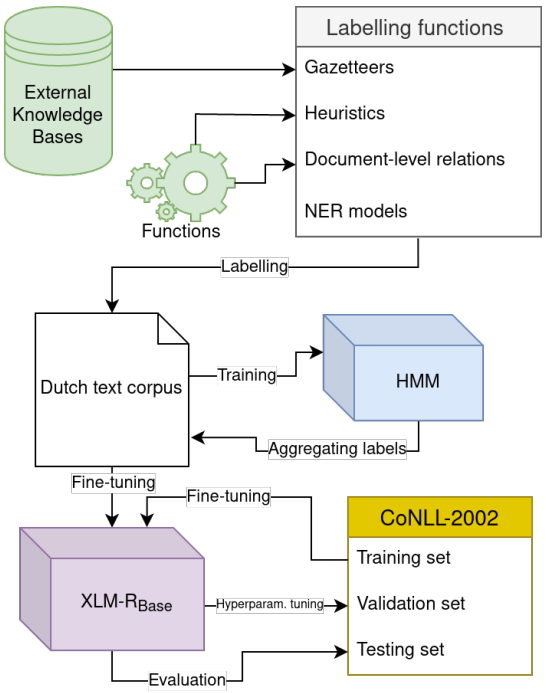
Named Entity Recognition for the Low-Resource Dutch Language

submitted in partial fulfillment for the degree of master of science

W.P.N. Bos
11333049

master information studies
data science
faculty of science
university of amsterdam
2021-07-07



	Internal Supervisor	External Supervisor	Second Examiner
Title, Name	Sara Mahdavi Hezavehi	Frank Smeets	Frank Nack
Affiliation	Universiteit van Amsterdam	Gemeente Amsterdam	Universiteit van Amsterdam
Email	s.mahdavihezavehi@uva.nl	frank.smeets@amsterdam.nl	F.M.Nack@uva.nl

Named Entity Recognition for the Low-Resource Dutch Language

William Bos
Universiteit van Amsterdam
Amsterdam, The Netherlands
wpnbos@gmail.com

Abstract

Recent political developments in the Netherlands and the European Union (EU) have imposed organisations with laws that restrict the way organizations can make use of user data through the General Data Protection Regulation (GDPR). Manual management of documents in context of the GDPR proves to be very labor intensive. Named Entity Recognition (NER) can be applied for the recognition of sensitive information in documents. NER for the Dutch language has seen promising improvements in recent years but due to the sensitive nature of privacy issues, improved performance is desirable before organizational deployment is feasible. Recent studies have shown that fine-tuning Pre-trained Language Models (PLMs) on Dutch NER specifically with more data improves performance. Due to the scarcity of quality labelled NER data in Dutch and other low-resource languages, this study seeks to improve NER performance for low-resource languages through the application of weak supervision. Results show that previous performance levels of English weak supervision approaches can be reciprocated for the Dutch language, and that fine-tuning using the resulting labelled data does not lead to performance improvements on a state-of-the-art PLM.

1 Introduction

Recent developments in both Dutch national and EU politics have seen the introduction of laws related to individual privacy. These laws provide constraints to organizations who work with user data as to provide the Dutch and EU citizens alike with an increased notion of privacy. Organizations working with personal data are constrained in their use of this data in both temporal and contextual domains. More concretely, the EU GDPR specifies that when processing personal information the organization must be explicit how the data will be used, that the data must used within the context and for the purpose it was initially collected for (contextual) [1], and that it must be deleted when this purpose is no longer applicable (temporal). According to the GDPR, personal information is data that can identify an individual, directly or indirectly [11]. Obvious examples are then first and last names, but less obvious examples are cookies and IP addresses.

Organizations have the liberty to comply with these laws in their own way, however not all methods are equally feasible. Data collections can span thousands of documents, all of which can be subject to dozens of different rules. Managing such a collection by hand is a laborious and time-consuming task. Therefore, automated solutions are desirable. In such an automated system dealing with personal information, the first step in processing it is locating the information. Once personal information has been located, multiple options exist for dealing with the information, i.e. it could be deleted or anonymized depending on legal requirements. The tagging of personal information (the automatic recognition of personal data such as names and addresses) in documents falls under the Named Entity Recognition (NER) section of Natural Language Processing (NLP). NER, as the name suggests, considers retrieving named entities from text; i.e. retrieving names and addresses from a building permit document. NER is typically applied through the use of Machine Learning (ML) models.

NER has recently seen significant improvements in performance levels by the introduction of Neural Networks (NNs), in particular through the introduction of transformer architectures by Vaswani and colleagues [23], and especially through the introduction of Bidirectional Encoder Representations from Transformers (BERT) by Devlin and colleagues [8]. These developments will be discussed more concretely later. Despite the successes of these recent NER innovations, reasons for improvement maintain.

In the context of GDPR privacy laws, any ML model performance imperfections mean vast amounts of privacy sensitive tokens are going unnoticed when considering deployment at large scale. This is not only a problem to organizations, which are prone to penalization by legislative authorities in the event of a GDPR breach, but also to the individual user who has a right to privacy. Even though state-of-the-art models achieve high scores already, which will be discussed in more detail later, these models would preferably return higher performance metrics as to preserve privacy as much as possible. Therefore, study seeks to improve state-of-the-art NER performance, for the purpose of facilitating Dutch organizations with privacy-preserving automated methods for information processing. Consequently, this study focuses on the Dutch language. As to explain the approach taken for improving the state-of-the-art, the next section details recent

innovations in transformer architectures which facilitated current state-of-the-art performance levels.

2 Background

2.1 BERT and derivations

Recently, the most significant innovation in transformer architectures came through BERT. Before BERT, transformer architectures were unidirectional in their processing of text, mostly processing text from left to right. This is sub-optimal for sentence-level tasks, as word context can only be observed in one direction. To remedy this, BERT was made to be bidirectional in its processing of text. Devlin and colleagues achieved this by applying the Masked Language Modelling (MLM) task during the training phase of the architecture. When performing a MLM task, a model has to predict what words were originally on the locations of now masked pieces of text, by considering the context. Additionally, BERT is trained using the Next Sentence Prediction (NSP) task, where the model predicts the next sentence given the current sentence and a set of options. This task is for example important for question answering NLP tasks [10].

After its initial release, BERT has seen revisions, both monolingual, similar to its initial English release, and multilingual. As this study concerns the Dutch language, recent developments in Dutch transformer models are discussed next. De Vries and colleagues [5] propose a monolingual version of the BERT architecture, which is trained on several Dutch corpora, called BERTje. BERTje achieves state of the art performance on several NLP tasks, but is outperformed on the NER task by mBERT on the CoNLL-2002 data set. The CoNLL-2002 data set [20] consists of both a Spanish and Dutch corpus, and has become the academic standard in bench marking NER methods for the Dutch language. The mBERT model is a multilingual revision of the BERT architecture developed by Devlin and colleagues [7], which is trained on 104 different languages.

2.2 RoBERTa and derivations

One notable English revision of BERT is RoBERT [15]. One major alteration to BERT proposed by Liu and colleagues is the removal of the NSP task in the pre-training of the model. Additionally, the study found that BERT as developed by Devlin et al. [8] was significantly undertrained, and that after sufficient training it could match or exceed the performance of every model published after BERT. In Delobelle et al. [6], authors developed a Dutch variant of the RoBERTa architecture, aptly named RobBERT. Contrary to expectations set by RoBERTa's impressive results, RobBERT does not achieve competitive scores for the NER task on the CoNLL-2002 set. The comparative analysis executed by Delobelle and colleagues find mBERT to be the best performing model on the NER task, much like de Vries and colleagues. However, XLM-R, a crosslingual alteration of the RoBERTa architecture developed by Conneau and colleagues [3], had not been released at this point. Conneau and colleagues find that XLM-R outperforms mBERT. Even though RobBERT has not directly been compared against XLM-R in the same experimental settings, these findings suggest that XLM-R outperforms robBERT too, as both models have been benchmarked on the same data set, albeit in different studies. Therefore, the best performing model at NER for the Dutch language currently is XLM-R, which achieves an F1 score of 92.53 on the CoNLL-2002 NER task in [3]. Additionally, a less resource demanding version of this architecture was released named XLM-R_{Base}. This architecture achieved an F1 score of 90.39. Recently, the XLM-R model has seen improved performance on this same task by the addition of document-level context to the model input [19]. This addition improved the F1 score to 93.99±0.16. The relevant metrics for all models discussed in this section are collected in table 1.

Model	F1
mBERT	90.94 [6]
BERTje	88.3 [6]
RobBERT	89.08 [6]
XLM-R_{Base}	90.39 [3]
XLM-R	92.53 [3]
XLM-R + context	**93.99** [19]

Table 1. NER models and F1 scores. Evaluated on CoNLL-2002 Dutch NER task.

2.3 NER fine-tuning

Models discussed so far represent the state of the art for NER, however these models have been pre-trained on objective functions different from an NER task. As mentioned, BERT models are subject to two different tasks during training, namely MLM and NSP, where RoBERTa based models have seen the NSP task dropped. Then when analysing performance for a specific task, these models are briefly fine-tuned on a task-specific training set and tweaked on the a validation set. Consequently, the models are evaluated on a test set. For most of Dutch NER related academic literature, the dataset referred to for NER task-specific gold standard labels is the CoNLL-2002 dataset (Tjong Kim Sang [20]). The training, development and test data sets contain 218737, 40656 and 74189 lines of words respectively. The training set has a size of 2.3MB uncompressed, where the corpus used for pre-training XLM-R contains 2.5TB of data sourced from the CCnet data set [24], several orders of magnitude more. Van Toledo and colleagues [22] find that when trained on the SoNAR-1 corpus [17], which contains 1 million words, BERT architectures perform better on a NER task on average than when trained on the CoNLL-2002 corpus, which contains only 218737 words. In the same vein, Schweter and Akbik

[19] find that when training XLM-R on not just the CoNLL-2002 training set, but also on the development set, which contains an extra 40656 (about 20 percent) lines improves the F1 score on the CoNLL-2002 dataset by 0.67±0.12, from 93.99 to 94.66. These results suggest that Dutch NER task performance can be improved by fine-tuning PLMs on increased amounts NER task-specific data. However, attaining task-specific data in low-resource languages such as Dutch is an issue, and manufacturing it manually is a laborious process. In such a scenario, one option is to opt for weak supervision.

2.4 Weak supervision

When utilizing weak supervision, data is labelled automatically, through the application of heuristics or by using a rule-based approach which can easily be automated. Weak supervision hails from the bio-informatics field, and was initially proposed by Craven and Kumlien [4]. When using external knowledge bases such as lists of names or locations for weak supervision, such an approach is called distant supervision [16]. By applying the knowledge of multiple external knowledge bases on the labelling of a data set, a voting procedure of sorts ensues where every knowledge base casts its vote as to what the true label should be for a target. Additionally, some knowledge bases might be specialized on a certain subset of labels. These knowledge bases, applied onto a corpus through labelling functions, have their labels aggregated into one predicted label. After all data has been processed by the aggregated labelling functions, a classification model can be trained on the newly labelled data. Lison and colleagues [14] have found that a NN model trained on this weakly labelled data is able to learn the domain expertise of the aggregated labelling functions, and even perform slightly better than the aggregated labelling functions.

2.5 Aggregating labelling functions

The procedure of aggregating labelling functions was generalized by Ratner and colleagues [18]. Results from studies using their work are promising, with Snorkel labelled data only being outperformed by hand labelled data sets by 3.6 percent on average on F1 scores. Snorkel allows for the programming of labelling functions in the programming language Python, and is publicly available [1]. However, as noted by Lison et al. [14], NLP and NER in particular operate by treating natural language as a sequence of words. Snorkel however requires input tokens to be independent, and therefore, is not suitable for NER tasks and consequently for this study. Lison et al. [14] resolve this problem by combining the knowledge of different labelling functions into one through the use of a Hidden Markov Model (HMM).

[1] See https://github.com/snorkel-team/snorkel

3 Research questions

This study builds on the work of Lison et al. [14] by using a HMM for the aggregation of labelling functions. This model will then be used to weakly label an unlabelled Dutch text corpus with NER tags. Consequently, this weakly labelled dataset will be used for fine-tuning the state of the art NER model XLM-R, for the purpose of determining the effect of weak supervision on state of the art NER models in the context of the Dutch language. Therefore, this study aims to answer the following research question (RQ):

> **RQ1.** *How does the fine-tuning of state-of-the-art pre-trained language models through weak supervision affect model performance on Dutch NER tasks?*

Model performance will be bench marked on the publicly available CoNLL-2002 Dutch NER data set. The study pipeline as described in this section is visualized in figure 1. For the purpose of determining the effect of weak supervision on model performance, weakly supervised data will have to be created, for which a model is required. As the performance levels of this weak supervision model are to be compared to previous research to establish model effectiveness, the following sub question will be answered:

> **Sub-RQ1.1** *With what performance levels can a weak supervision model label Dutch NER data?*

By answering the sub research question, this study aims to clarify if methods for weak supervision developed by Lison et al. [14] can have their results reproduced for other low-resource languages by attempting to do so for the Dutch language. Reproducing their results for a different target language than English would additionally validate the methods of Lison et al. [14]. Furthermore, the answer to the main research question will show if low-resource languages with already high performance metrics for NER models can benefit from weak supervision, and potentially provide the scientific community with a low-cost method for improving state-of-the-art NER models.

4 Theoretical background

For the purpose of illuminating some otherwise unfamiliar terms, the next section details some concepts which are key to this study.

4.1 Labelling functions

Labelling functions are code functions which utilize either a knowledge base for distant supervision, or encoded logic for the purpose of weak supervision. An example of the latter, a simplified labelling function for recognizing first names is illustrated in algorithm 1. This example checks every token in a document for matches with a list of first names, and upon a match produces the label PERSON. Although the labelling functions applied in this study are more sophisticated in

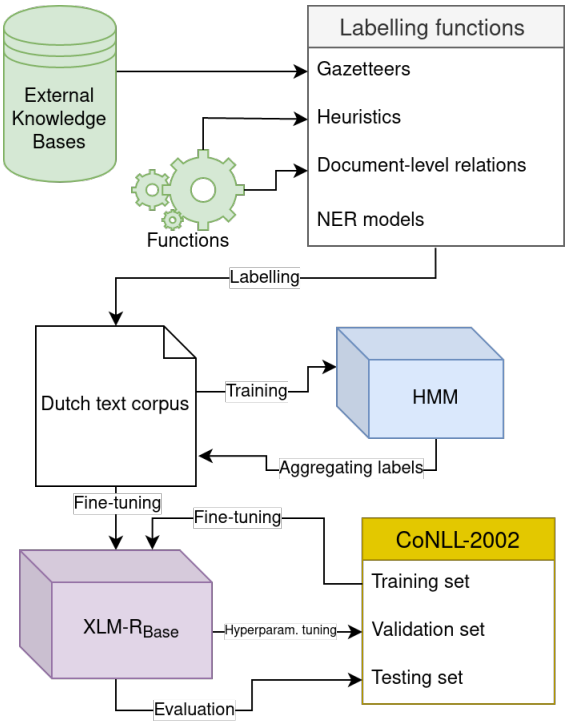


Figure 1. Pipeline diagram

their logic, this example shows the basic structure of most labelling functions. After all labelling functions have been applied to all documents, their "votes" are aggregated by a HMM.

Algorithm 1: Labelling function example

```
Data:
document a document with at least one token
first_names a list of first names i.e. {Alice, Bob, ...,
Zachary}
Result: document with labelled first names
for token ∈ document do
    if token ∈ first_names then
        label token as PERSON;
    else
        label token as NO ENTITY;
    end
end
```

4.2 Hidden Markov Models

Within probability theory, Markov models are used to model changing systems where future states are assumed to depend only on the current state. Within a Markov model, the state of a system is ought to be observable. An example of such a system is the weather: one can observe whether the sun shines or if the weather is rainy. A Markov model then models the likelihood of transitions between states, given the current state: how likely is it that the next state of the weather is rainy given that it is sunny now?

A Hidden Markov model is employed in situations where the current state of the system is not fully observable. An example of such a system would be the weather system as described above, but instead of being able to see whether if it is sunny or rainy, one can only see the way people are dressed: people outside have umbrellas, or people are dressed in shorts. People wearing shorts while it is sunny is called an *emission* of a state. Emissions are observations related to the state of the system, but are insufficient for precisely determining system state: some people might wear shorts while it is raining.

As mentioned before, the classification of text tokens comes with a challenge: tokens cannot be assumed to be independent when located in a sentence. Words do not exist independently in a sentence, and influence each other's meaning. As previous frameworks for weak supervision depend on the independence of tokens (Ratner et al. [18]), an alternative solution was developed by Lison et al. [14], which has seen further development into the Skweak framework (Lison et al. [13]). This alternative solution employs a HMM for the aggregation of labels. In this HMM, the individual labelling functions are emissions from the state. The state, being hidden, is the true label of the token. The learnable

parameters for this HMM are then the transition matrix between the states, and the probabilities of emissions from the labelling functions given a state.

5 Methodology

This study can be partitioned into two parts: the establishing of a weakly labelled data set, and the utilization of this data set for fine-tuning a state-of-the-art NER model. Each part answers a research question, with the first part answering the sub research question, and the latter answering the main research question. In global terms, the first part establishes a weakly supervised data set through the construction of a HMM by aggregating labelling functions' "votes". A more in-depth method follows.

For the purpose of acquiring knowledge to encode into labelling functions, a set of external knowledge bases from various sources, which are discussed later, is collected. Consequently, these external knowledge bases are encoded into labelling functions. Once these labelling functions have been developed and have had their votes applied to the target corpus, the labelling functions' "votes" are aggregated through the use of a HMM, the process of which is abstracted through the Skweak library. The result is a single entity label for every token in the target corpus: weakly supervised data.

Consequently, the labelling model is applied to an unlabelled Dutch text corpus for the purpose of generating a large quantity of weakly supervised training data. For this study, the website of the Dutch public broadcaster NOS has been scraped for historic news articles in Dutch. The choice to scrape news articles was made because ConLL-2002 is constructed of news articles. Furthermore, as NOS is a public broadcaster, their documents are deemed as more easily available in the public domain than proprietary news sources and therefore more fitting to scientific principles.

The resulting weakly labelled NOS corpus is used for the purpose of fine-tuning the XLM-R model, after which the final round of evaluation will occur. For the purpose of this final evaluation, three experiments are conducted, each of which its result is compared to the baseline of XLM-R fine-tuned on the CoNLL-2002 training set. These experiments are described in section 7.3.

6 Experimental setup

All experiments conducted in this study took place in a Google Colaboratory [2] environment. All final benchmarks were produced by a Tesla T4 GPU. Due to the computational constraints inflicted on this study by the limits of this GPU, it was not possible to replicate the state-of-the-art baseline benchmark for the full XLM-R model as it was too large. Therefore, this study shifted from using XLM-R to using XLM-R_{Base}, which is a XLM-R version about half the size of the large model, with 250 million parameters for XLM-R_{Base} versus 560 million parameters for XLM-R. Any results forthcoming from this study will therefore not be directly sourced from the state-of-the-art XLM-R. However, as XLM-R_{Base} is simply a shrinked version of the state-of-the-art model, results of experiments of the base model are deemed to likely be representative of potential results from the larger model. As to detail the execution of the methodology described above, the next section describes the experimental setup, starting with the data sets used and details about their processing.

6.1 Datasets

CoNLL-2002. The CoNLL-2002 dataset produced by Tjong Kim Sang [20] will be utilized for its Dutch partition. The Dutch partition contains four label categories, persons (PER), organizations (ORG), locations (LOC) and miscellaneous names (MISC). These are consequently the labels that the model in this study will be evaluated on. The label distributions for each partition of this data set are collected in table 2, along with the partition sizes. For every label, the occurrences of the label are shown for every data split, along with the percentage of all labels that these occurrences make up in that particular data split. The bottom row details total partition sizes, and the percentages of the total data set that these data splits make up.

The CoNLL-2002 dataset[2] was sourced in the predetermined training, validation (development) and test sets. The partitions used in this study are the partitions provided by the original conference. The data is available in IOB format, where every line is a token and token labels are separated from the token and other labels by whitespace. For utilization in this study, the IOB format files were converted to Spacy's Docbin format for faster processing and compatibility with Skweak [13].

NOS corpus. For the purpose of acquiring unlabelled Dutch documents, the website of the Dutch public broadcaster NOS was scraped for news articles, for aforementioned reasons. The scraping was done by utilizing the Wayback Machine[3] for its archive of the NOS RSS feed. The corpus resulting from the scraping contains 1000682 tokens in 5897 documents, sourced from between 2012 and 2021, and is therefore about five times the size of the CoNLL-2002 training set. These documents were stripped of any markup language tags, resulting in raw text consisting of only natural language.

6.2 Weak supervision

The creation of the HMM weak supervision model can be summarized into three parts:

1. the collection of external knowledge bases;

[2]Sourced from https://www.clips.uantwerpen.be/conll2002/ner/
[3]See https://archive.org/web/

Partition	Train	Validation	Test
B-PER	4716 (24.4%)	703 (18.9%)	1098 (19.0%)
I-PER	2883 (14.9%)	423 (11.4%)	807 (14.0%)
B-LOC	3208 (16.6%)	479 (12.9%)	774 (13.4%)
I-LOC	467 (2.4%)	64 (1.7%)	49 (0.85%)
B-ORG	2082 (10.8%)	686 (18.5%)	882 (15.3%)
I-ORG	1199 (6.2%)	396 (10.7%)	551 (9.6%)
B-MISC	3338 (17.3%)	748 (20.1%)	1187 (20.6%)
I-MISC	1405 (7.3%)	215 (5.8%)	410 (7.12%)
All	19298 (67.1%)	3714 (12.9%)	5758 (20.0%)

Table 2. Class label distributions across ConLL-2002 partitions.

2. the coding of labelling functions, such as encoding external knowledge bases into functions and the encoding of heuristics into code;
3. the application of labelling functions onto a target corpus and subsequent aggregation and model training.

A detailed description of the proceedings for these parts follows.

(1) A set of knowledge bases used in Lison et al. [14] was acquired and translated to Dutch. Additionally, the Geonames geographic database was consulted for a list of locations specific to the Netherlands. The knowledge bases, after translation, were stored in JSON format where the keys represented a label, such as PER or ORG, and the values a list of entities corresponding to the labels.

(2) the coding of the labelling functions was done in Python, utilizing the Skweak package for transforming ordinary Python functions into labelling functions capable of being used for aggregation. The encoded labelling functions can be categorized into four categories as defined by Lison et al. [14], with four distinct approaches to encoding them: NER models, gazetteers, heuristic functions and document-level relations.

NER models in this study consist of trained sequence labelling models, specifically trained for the purpose of NER. In this study, three such models are utilized: a model trained on UD Lassy Small (Van Noord et al. [21]), a model trained on ConLL-2002 (Tjong Kim Sang [20]), and a model trained on SONAR-1 (Oostdijk et al. [17]). These models are incorporated in the labelling function collection through calling these models in inference mode upon a sequence of tokens, and using the predicted labels as "votes".

Gazetteers represent the encoded external knowledge bases. The external knowledge bases represented are sourced from the Crunchbase Open Data Map, Geonames[4], and Wikipedia (Geiß et al. [9]). In short, gazetteers are collections of known entities with their labels. A gazetteer function then looks for any matches in its knowledge base found in the text to be processed, and produced labels where necessary.

Heuristic functions are functions that apply rule-based knowledge in the form of heuristics (or "rules of thumb" [12]) to the target text. And example of a heuristic function is the function in algorithm 1. These functions aim to capture logic and common patterns in natural language, and allows for the capturing of entities otherwise undetected by NER models or gazetteers.

Document-level relations capture the relations between similar entities within a document. When for example an article mentions a person for the first time, the mention usually takes shape as a full name, whereas later in the document the mention might be abbreviated to just a first or last name. Document-level relations then aim to recognize the occurrence of this last name later in the document as a person, rather than an organization or location. Some other examples of document-level relations, both implemented in Skweak Lison et al. [13], are a majority voter on the function level, which replicates the vote for a particular entity which holds the majority of all votes from other labelling functions, and the document history majority voter which looks at earlier classifications of the same entity in a document and votes for the label that appears most often for that particular entity.

To determine the amount of labelling functions required, inspiration was taken from Lison et al. [14], who wrote 52 labelling functions for their multi-class problem consisting of 9 classes, resulting in 5.7 labelling functions per class. However, many of their labelling functions are cased/uncased versions of the same function, or single/multi-token implementations of the same labelling function. As to increase the odds of reciprocating their results in this study for a different target language, this study developed a similar ratio of labelling functions to classes, resulting in a total of eighteen labelling functions: 4.5 functions per class on average. Empirical experiments were done with more labelling functions although none were found to improve the eighteen function model.

(3) Through use of the Skweak package, labelling functions had their votes cast on every document in the training set of ConLL-2002. Once these labels were applied, a HMM was fitted on the noisy labels, resulting in a trained model and a single label for each entity produced through aggregation by the model. Following the creation of this weak supervision model came its application to the unlabelled NOS corpus and subsequent experiments using this data for training a state-of-the-art NER model.

[4]See https://www.geonames.org/

6.3 Experiments

The previous section of this study resulted in a weakly labelled NOS corpus consisting of about one million tokens. The next section describes the experiments conducted using this data. First, XLM-R_{Base} was trained on the ConLL-2002 training set for the purpose of validating and reproducing the benchmark of Conneau et al. [3]. For the purpose of training the transformer model and using it for inference with relative ease, the Transformers Python library by Wolf et al. [25] was used. After this baseline was confirmed, as mentioned previously, three experiments were conducted: For the first experiment, the weakly labelled NOS corpus is used as fine-tuning data for XLM-R. For the second experiment, the same weakly labelled NOS corpus is used as fine-tuning data, but for an XLM-R model already fine-tuned on ConLL-2002. Finally, for the third experiment ConLL-2002 training data and weakly supervised NOS data are combined into one set on which XLM-R is then fine-tuned. Every experiment lasted five epochs, as validation performance decreased after this point. Additionally, out of two, four, eight, and sixteen, a batch size of eight was found to be optimal. Larger batch sizes have not been experimented with due to the computational constraints described before.

6.4 Evaluation

The HMM resulting from the sub research question was evaluated by estimating CoNLL-2002 test set labels, after which precision, recall and F1 metrics were calculated given the estimated and true labels. As to contextualize final results, weak supervision accuracy will be compared to results from Lison et al. [14] for the English language. It has to be noted that Lison and colleagues evaluated their model's performance on ConLL-2003's English partition. To compare results between two different languages might be questionable. However, as research at time of writing has not yet been executed using their framework for Dutch, their weak supervision performance for English is deemed as the most suitable baseline available to this study. Additionally, ConLL-2003's data set is from the same conference and contains the same labels, which aids comparability. As for the XLM-R experiments conducted for the purpose of answering the main research question, for each of the three experiments the precision, recall and F1 performance on the ConLL-2002 test set are computed as well. Following the evaluation, metrics will be compared to baseline XLM-R performance on the ConLL-2002 test set as established by Conneau et al. [3].

7 Results

Sub-RQ1.1. *With what performance levels can weak a supervision model label Dutch NER data?* The results produced while answering the sub research question are shown in table 3. The metrics are micro-averaged. Additionally, metrics are shown for different subsets of labelling functions.

Model	P	R	F1
Lison et al. [14] (ConLL-2003 English)	0.719	0.794	0.754
HMM-aggregated labels (all functions)	0.788	0.779	**0.783**
HMM-aggregated labels (only NER models)	0.780	**0.782**	0.781
HMM-aggregated labels (only gazetteers)	0.478	0.255	0.333
HMM-aggregated labels (only heuristics)	**0.831**	0.198	0.320
HMM-aggregated labels (all but doc-level)	0.790	0.768	0.779

Table 3. Evaluation results on ConLL-2002 test set.

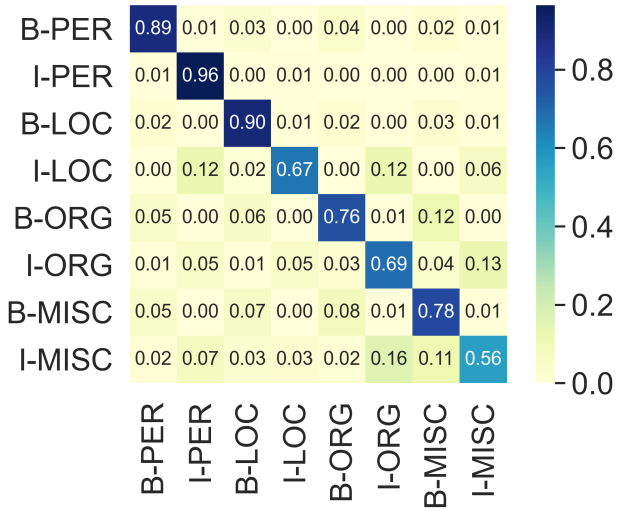


Figure 2. Weak supervision confusion matrix.

Observations include that the best F1 score is observed for the HMM model that utilizes all functions, but that the difference between the *all functions* model and the *only NER models* is minimal. To concretely answer the sub research question, performance levels of weak supervision using a HMM from Lison et al. [14] for English have been reciprocated for Dutch, although, as noted before, the results cannot be directly compared due to different test sets. Figure 2 shows the confusion matrix for the HMM performance. Sizable differences between classes are noticeable, with the highest accuracy belonging to the PER classes, and the lowest accuracy to the MISC classes. Implications of the findings in this subsection are further discussed in the discussion.

RQ1. *How does the fine-tuning of state-of-the-art pre-trained language models through weak supervision affect model performance on Dutch NER tasks?* The results for the main research question, the results from the XLM-R_{Base} experiments, are shown in table 4. Like the results for the sub research question, the metrics are micro-averaged. Precision, recall and F1 scores are not available for the baseline from Conneau et al. [3] as they are not provided in their study. The immediate observation to make is that none of the three experiments

	P	R	F1	CEE
ConLL-2002 (Conneau et al. [3])	-	-	**90.39**	-
NOS	78.37	74.75	76.52	0.210
ConLL-2002 + NOS (sequential)	79.67	78.58	79.67	0.094
ConLL-2002 + NOS (combined)	87.12	86.41	86.77	0.056

Table 4. Results for XLM-R_{Base} fine-tuning experiments.

resulted in higher scores than the baseline model. Additionally, the model performed significantly better when receiving ConLL-2002 training data and NOS data in combined fashion rather than when in sequential fashion. The answer to the research question is then that the fine-tuning of a state-of-the-art pre-trained language model through weak supervision negatively affects performance on Dutch NER tasks for the weak supervision as applied in this study. The implications of the results for this question, too, are discussed further in the discussion.

8 Discussion

Reflecting upon table 3, the fact that the difference in performance metrics between the *all-functions* model and the *only NER models* model is minimal raises the issue of the cost-benefit relationship between the two approaches. The *only NER models* model consists of a mere three labelling functions, whereas the complete model consists of six times more functions, all of which raise computational costs and processing times. The choice for either model will depend on contextual factors, as some contexts might not deem additional performance at such costs worthwile. Lison et al. [14] found labelling functions which capture document-level relations to be performance enhancing for weak supervision, and the same conclusion can be drawn from this study, as the *all but doc-level* model performs worse than the *all functions* model.

The experiments conducted in this study used eighteen labelling functions, the majority of which were adapted from Lison et al. [14] and Lison et al. [13]. The confusion matrix as seen in figure 2 show that for some classes, the labelling functions employed in this study perform better than for others, with performance differences being sizable. Thus, developing additional labelling functions for these classes could yield improved performance. Additionally, on average across classes, the *CLASS*-I labels see worse performance than the *CLASS*-B labels. Therefore, future research could see improved performance by focusing on better recognition of multi-token entities. Regarding the confusion matrix, the imbalance in performance across class labels shows traits favorable to optimizing privacy with NER as the most privacy sensitive labels (the PER labels and the B-LOC label) show the best performance.

The experiments conducted with XLM-R_{Base} show that the weakly supervised data as produced in this study is not usable for developing a state-of-the-art NER model. The likely explanation is that the accuracy of the weakly supervised labels is not sufficient for improving the state-of-the-art. An interesting observation is, however, that the way data is fed to the network during fine-tuning does quite heavily impact performance measures: the sequential mode of fine-tuning produced decisively lower performance metrics than the combined mode. Additionally, the difference in sizes between the ConLL-2002 dataset and the NOS dataset (the latter being five times larger) would lead one to expect the combined scenario performance to be closer to the NOS-only metrics, but results show otherwise. An explanation for these phenomena is that in the sequential scenario, the model optimizes in completely separate fashion for both data sets, with the last optimization being from the out-of-domain NOS data set. In doing so, the model "forgets" the beneficial patterns learned in from the gold-standard labels from ConLL-2002 in favor of learning detrimental patterns from the lower accuracy labels provided by the weakly labelled NOS data set. In the combined scenario, perhaps the model learns to emphasize the samples where both data sets agree, therefore creating a model much closer to the baseline model.

Results as presented in this study do not exclude the potential usefulness of weak supervision for developing state-of-the-art models. Even though Dutch is a low-resource language, Dutch NER performance metrics are amongst the highest for any language for this classification task. Low-resource languages where NER models are not as developed yet could see reward from deploying weak supervision methods as used in this study.

On a more meta-scientific note, evaluating performance metrics for NER is typically done using the F1 score, which represents a rough mid-point between precision and recall. However, when NER is put into practice with the goal of preserving privacy, the question has to be raised if precision should weigh as heavily as recall. When privacy is the main goal, one could imagine that being overly cautious and tagging entities with little restraint would optimize performance towards this goal: it is better to be safe than sorry. In such a scenario, precision scores will suffer, and consequently, so will F1 scores, even though the privacy-enhancing objective is met. However, the abolishment of precision's influence in the F1 score would yield a model simply tagging every token as a named entity. Future research regarding NER for optimizing user privacy should look into adapting the F1 score for a more privacy-sensitive objective, by perhaps tweaking the weight both recall and precision have on the final score.

9 Conclusion

This study reproduced efforts by Lison et al. [14] for a different, low-resource target language by applying weak supervision methods using a HMM tested on English to the Dutch language. Labelling functions were developed to automate the labelling of unsupervised data sets, with each function having an own area of expertise. Consequently, labels were aggregated, producing weakly labelled data. Results for weak supervision applied this way show that an eighteen labelling function model barely outperforms a model of three NER models encoded into labelling functions. However, similar scores as the ones established by Lison et al. [14] for English were observed for Dutch testing data. The reproduction of HMM-enabled weak supervision was executed for the purpose of generating more fine-tuning data for training a state-of-the-art NER model, for a low-resource language. Results show that the weakly labelled data produced in this study does not enable the production of a state-of-the-art model but rather decreases performance due to label inaccuracies. Even though the results of this study did not yield a state-of-the-art model for Dutch NER, results did show that weak supervision through HMM label aggregation can be reproduced across languages and as such the findings of this study can provide valuable for other low-resource languages, particularly languages with minimal resources for NER. Future work should look into improving upon the selection of labelling functions deployed in this study, with lesser performing classes such as MISC and ORG, and multi-token entities deserving most attention.

References

[1] David Basin, Søren Debois, and Thomas Hildebrandt. 2018. On purpose and by necessity: compliance under the GDPR. In International Conference on Financial Cryptography and Data Security. Springer, 20–37.

[2] Ekaba Bisong. 2019. Google colaboratory. In Building Machine Learning and Deep Learning Models on Google Cloud Platform. Springer, 59–64.

[3] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116 (2019).

[4] Mark Craven, Johan Kumlien, et al. 1999. Constructing biological knowledge bases by extracting information from text sources.. In ISMB, Vol. 1999. 77–86.

[5] Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch bert model. arXiv preprint arXiv:1912.09582 (2019).

[6] Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. Robbert: a dutch roberta-based language model. arXiv preprint arXiv:2001.06286 (2020).

[7] Jacob Devlin. 2018. Multilingual Bert readme. https://github.com/google-research/bert/blob/master/multilingual.md [Online; accessed 10-March-2021].

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).

[9] Johanna Geiß, Andreas Spitz, and Michael Gertz. 2017. Neckar: A named entity classifier for wikidata. In International Conference of the German Society for Computational Linguistics and Language Technology. Springer, 115–129.

[10] Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. 2016. Contextual lstm (clstm) models for large scale nlp tasks. arXiv preprint arXiv:1602.06291 (2016).

[11] Michelle Goddard. 2017. The EU General Data Protection Regulation (GDPR): European regulation that has a global impact. International Journal of Market Research 59, 6 (2017), 703–705.

[12] Douglas B Lenat. 1982. The nature of heuristics. Artificial intelligence 19, 2 (1982), 189–249.

[13] Pierre Lison, Jeremy Barnes, and Aliaksandr Hubin. 2021. skweak: Weak Supervision Made Easy for NLP. arXiv preprint arXiv:2104.09683 (2021).

[14] Pierre Lison, Aliaksandr Hubin, Jeremy Barnes, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. arXiv preprint arXiv:2004.14723 (2020).

[15] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019).

[16] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 1003–1011.

[17] Nelleke Oostdijk, Martin Reynaert, Véronique Hoste, and Ineke Schuurman. 2013. The construction of a 500-million-word reference corpus of contemporary written Dutch. In Essential speech and language technology for Dutch. Springer, Berlin, Heidelberg, 219–247.

[18] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases, Vol. 11. NIH Public Access, 269.

[19] Stefan Schweter and Alan Akbik. 2020. FLERT: Document-Level Features for Named Entity Recognition. arXiv preprint arXiv:2011.06993 (2020).

[20] Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002). https://www.aclweb.org/anthology/W02-2024

[21] Gertjan Van Noord, Gosse Bouma, Frank Van Eynde, Daniel De Kok, Jelmer Van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. 2013. Large scale syntactic annotation of written Dutch: Lassy. In Essential speech and language technology for Dutch. Springer, Berlin, Heidelberg, 147–164.

[22] Chaïm van Toledo, Friso van Dijk, and Marco Spruit. 2021. Dutch Named Entity Recognition and De-identification Methods for the Human Resource Domain. arXiv preprint arXiv:2106.02287 (2021).

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. arXiv preprint arXiv:1706.03762 (2017).

[24] Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzman, Armand Joulin, and Edouard Grave. 2019. Ccnet: Extracting high quality monolingual datasets from web crawl data. arXiv preprint arXiv:1911.00359 (2019).

[25] Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. 38–45.