École Polytechnique Fédérale de Lausanne

Boosting named entity recognition in domain-specific and low-resource settings

Doctoral semester project

Written by Sven Najem-Meyer under the supervision of Matteo Romanello and Maud Ehrmann

13th January 2022

Contents

Abstract. Recent researches in natural language processing have leveraged attention-based models to produce state-of-the-art results in a wide variety of tasks. Using transfer learning, generic models like BERT [1] can be fine-tuned for domain-specific tasks using little annotated data. In the field of digital humanities and classics, bibliographical reference extraction counts among the domain-specific tasks where few annotated datasets have been made available. It therefore remains a highly challenging Named Entity Recognition (NER) problem which has not been addressed by the aforementioned approaches yet. In this study, we try to boost bibliographical reference extraction with various transfer learning strategies. We compare three transformers to a Conditional Random Fields (CRF) developed by Romanello [2], using both generic and domain-specific pre-training. Experiments show that transformers consistently improve on CRF baselines. However, domain-specific pre-training yields no significant benefits. We discuss and compare these results in light of comparable researches in domain-specific NER.

1 Introduction

1.1 Background

Named entity recognition Named entity recognition (NER) is a sequence labelling task which aims at detecting and at classifying entities in texts, according to a given typology. Typologies generally include entity types such as persons, organizations and locations [3, 4], but the same techniques can be applied to extract more specific entity types such as chemicals [5], diseases [6] or, as in the present study, bibliographical entities. Serving as a groundwork to entity linking and relation extraction, NER is deemed a major task in information extraction.

Transfer learning Like many other fields in natural language processing (NLP), NER has greatly benefited from deep neural networks, and is further benefiting from recent advances in transfer learning and attention-based models. Transfer learning consists in training a model $\mathcal{M}$ for a source task $\mathcal{T}_S$ in a source domain $\mathcal{D}_S$, before adapting it to a target task $\mathcal{T}_T$ and its corresponding domain $\mathcal{D}_T$. For the clarity of following sections, we introduce the notation proposed by Pan and Yang [7]. We first define a domain $\mathcal{D}$ by a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, where $X = \{x_1, x_i, ..., x_n\} \in \mathcal{X}$. Here, x_i is the ith feature in X and can itself be vector. For example, if features represent words as m-dimensional vectors (or "embeddings"), $\mathcal{X}$ corresponds to the total vocabulary and X to a sample of embeddings $\{x_1, x_i, ..., x_n\}$ with $x_i \in \mathbb{R}^m$. We then define a task $\mathcal{T}$ performed on $\mathcal{D} = \{\mathcal{X}, P(X)\}$ by a label space $\mathcal{Y}$ and a predictive function or model $\mathcal{M}$. In the case of NER, a possible scenario is $\mathcal{Y} = \{Person, Organization, Location, O\}$ and $\mathcal{M}(X) = P_{\mathcal{M}}(Y|X)$ where $Y = \{y_1, y_i, ..., y_n\} \in \mathcal{Y}$. Any predictive function that maps a conditional probability distribution $P(Y|X)$ to a sample X is a statistical model. It is defined as $\mathcal{M} = \{C, \Theta\}$, where C denotes a configuration which determines the model's architecture and hyperparameters, and where $\Theta = \{\theta_1, ..., \theta_n\} \in \mathbb{R}$ is a set of trainable parameters. Given a configuration C, a task $\mathcal{T}$ and a sample of labelled training data $\{X_{\mathcal{T}}, Y_{\mathcal{T}}\}$, training $\mathcal{M}$ corresponds to optimizing Θ so that the confusion between the predictions $\hat{Y}_t = \mathcal{M}(X_t)$ and the actual labels Y_t is minimized. Given source and target domains and tasks $\mathcal{D}_S$, $\mathcal{T}_S$, $\mathcal{D}_T$ and $\mathcal{T}_T$, the goal of transfer learning is to facilitate the training of a target model $\mathcal{M}_T$ on $\mathcal{D}_T$ for $\mathcal{T}_T$, by transferring the knowledge acquired by a source model $\mathcal{M}_S$ trained on $\mathcal{D}_S$ for $\mathcal{T}_S$.

Transferring knowledge from one model to the other implies that at least a subset of $\Theta_{\mathcal{M}_S}$ is kept in $\mathcal{M}_T$. This procedure generally requires three steps, which are usually referred to as pre-training, model adaptation and fine-tuning. Pre-training consists in training $\mathcal{M}_S$ for $\mathcal{T}_S$ on $\mathcal{D}_S$. As transfer learning naturally supposes $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$, it is paramount to chose $\mathcal{D}_S$ and $\mathcal{T}_S$ so that the $\mathcal{M}_S$ gains meaningful, generic and hence transferable representations

from pre-training. In NLP, language modelling appears to be the most appropriate task to learn such representations. The goal of the task is to predict a token given its context. As tokens can be automatically selected from their contexts to provide $\{X = context, Y = token\}$ samples, language modelling requires no manual labeling of the training data. Hence, vast amounts of training data can easily be created from publicly available corpora. In order to produce generally transferable representations, state-of-the-art models like BERT [1] are pre-trained for language modelling on vast corpora of generic texts, such as newspapers or Wikipedia articles.

Once $\mathcal{M}_S$ is trained, the second step of transfer learning is model adaptation. It consists in adapting $\mathcal{M}_S$ – and more specifically C_S – to $\mathcal{T}_T$. This step is crucial as $\mathcal{Y}_S$ and $\mathcal{Y}_T$ may have different dimensions. Indeed, in order to predict a token for language modelling, $\mathcal{M}_S$ is generally expected to output a softmax vector of dimension $|\mathcal{Y}_S| \approx |\mathcal{X}_S|$. For NER however, the model is expected to output a softmax of dimension $|\mathcal{Y}_T|$, which equals the numbers of entity types. It is therefore requisite to adapt $\mathcal{M}_S$'s output layers to fit $\mathcal{T}_T$. In practice, a subset of the trainable parameters $\Theta_{\mathcal{M}_S}$ is transferred and serves as a basis for building the adapted $\mathcal{M}_T$. In the case of encoder-decoder architectures, the encoder of the language model $\mathcal{M}_S$ is usually kept. The decoder however, is replaced by a $\mathcal{T}_T$-tailored decoder on top of it. Finally, the third and last step of transfer learning is commonly known as fine-tuning. It consists in training $\mathcal{M}_T$ for $\mathcal{T}_T$ on $\mathcal{D}_T$. It is worth noticing that during fine-tuning, one can choose not to update the part of $\mathcal{M}_T$ composed by $\Theta_{\mathcal{M}_S}$, a strategy called freezing.

Pre-training has shown a beneficial impact and can be performed with various language modelling strategies, such as next token prediction [8], previous token prediction (bi-directional approaches) and masked token prediction [1], skipgrams [9] and negative sampling [10]. As different as they may appear, these language modelling strategies all rely on the distributional hypothesis, which stipulates that similar words appear in similar contexts, and are hence to be given similar representations by the model. This principle was first used to generate static word representations such as Word2Vec [9] or GloVE [10]. Starting from a predefined vocabulary (corresponding to $\mathcal{X}_S$), Mikolov et al. [9] trained a log-linear model to predict a randomly selected token given another randomly selected token appearing in the same text window. The trainable parameters of this simple model can be construed as representations of tokens. They can serve as a basis for building a target model $\mathcal{M}_T$. These static embeddings have two main limitations though. First, they rely on a predefined vocabulary which corresponds to the feature space $\mathcal{X}_S$. This implies that words present in $\mathcal{X}_T$ but absent from $\mathcal{X}_S$ have no meaningful representations. Secondly, when used for a target task $\mathcal{T}_T$, static embeddings are not context-aware. In other words, the embedding of the word "mouse" is processed in the same way in the context of "rodents" and of "keyboard". This can create serious limitations when contexts and distributions significantly vary between $\mathcal{D}_S$ and $\mathcal{D}_T$.

Attention More recent approaches address these issues with the use of attention-based models. Attention is a weighting mechanism introduced by Bahdanau et al. [11] which builds on the encoder-decoder architecture [12]. The blueprint of this architecture is to have an encoder *encode* an internal representation of a sequence which is then used at every time step as input by the decoder. This is particularly appropriate in machine translation, as it allows the decoder to constantly consider the original sequence when generating the translated sequence. Instead of focusing on a single encoder representation (e.g. the first or last time step's representation), Bahdanau et al. [11] proposed to use all the internal representations of the encoder as input to the decoder and to train a separate mechanism that dynamically weights the importance of each representation at each time step. Given a sequence of internal states $S_{int} = (s_{int}^{<1>}, ..., s_{int}^{<T>})$, the attention matrix c is trained to optimize $\sum_1^T S_{int}^{<i>} * c_i$ to predict $Y_S^{<i>}$. This way, the model learns to shift its

"attention" to relevant parts of the source sequence. As this approach showed encouraging results, Vaswani et al. [13] proposed a new encoder-decoder architecture, the transformer, which relies only on attention. Representations from transformers are computationally effective, dynamic and context aware, as each token in sequence is always represented as a weighted sum of its context.

As mentioned above, transformers like BERT [1], RoBERTa [14] or DistilBERT [15] are pre-trained for language modelling on vast generic corpora such as Wikipedia. The resulting source model $\mathcal{M}_S$ can then be adapted to $\mathcal{T}_T$ by replacing language modelling decoder with a $\mathcal{T}_T$-specific decoder. Hence, the "pre-trained" part of transformer model designates the encoder of its source language model $\mathcal{M}_S$. In order for a pre-trained model to be used for NER, a softmax or a conditional random fields (CRF) decoder is commonly added on top of the encoder. The obtained target model $\mathcal{M}_T$ can then be fine-tuned for $\mathcal{T}_T$ with a relatively small labelled dataset. In NER, this strategy allowed BERT to reach an F1 score of 92.08 on the ConLL-2003 benchmark [3], a dataset commonly used to assess performances in NER. This result placed BERT largely above traditional machine learning methods like CRF[1].

1.2 Domain and task specificity

Domain specificity As mentioned above, transfer learning supposes that $\mathcal{D}_S \neq \mathcal{D}_T$ or that $\mathcal{T}_S \neq \mathcal{T}_T$. These conditions give rise to various scenarios. The first possible scenario is domain specificity, i.e. $\mathcal{D}_S \neq \mathcal{D}_T$. Since $\mathcal{D} = \{\mathcal{X}, P_S(X)\}$ domain specificity occurs if $\mathcal{X}_S \neq \mathcal{X}_T$ or $P_S(X) \neq P_T(X)$. The first part of the disjunction accounts for a difference between source and target feature spaces. This occurs when the source vocabulary $\mathcal{X}_S$ on which $\mathcal{M}_S$ is trained differs from the target vocabulary $\mathcal{X}_T$ to which $\mathcal{M}_T$ is confronted to. This leads to out-of-vocabulary issues, which are traditionally addressed by adding a placeholder feature for unknown words or by mixing character, chunk and word embeddings. This way, words composed of similar sub-pieces are be processed in a comparable way. The second part of the disjunction represents the case where the probability distribution varies between $\mathcal{D}_S$ and $\mathcal{D}_T$. This happens with changes in content, genre, style and writing conventions, so in bio-medical articles, legal documents or publications in classical studies. Technical jargon, if not completely absent from generic corpora, can still appear much more frequently and be distributed in thoroughly different contexts. For instance, terms like *catharsis* or *hybris* are frequent in classical studies but rather scarce in generic corpora.

Task specificity The second possible scenario is task specificity i.e. $\mathcal{T}_S \neq \mathcal{T}_T$. As a task is defined by $\mathcal{T} = \{\mathcal{Y}, \mathcal{M}\}$, task specificity implies either $\mathcal{Y}_S \neq \mathcal{Y}_T$ or $\mathcal{M}_S \neq \mathcal{M}_T$. Here, we focus on the first part of the disjunction. Given two tasks $\mathcal{T}_1$ and $\mathcal{T}_2$ with similar label spaces but different domains, one could consider doing transfer learning from $\mathcal{T}_1$ to $\mathcal{T}_2$. This procedure is sometimes referred to as task-tuning [16]. Its goal is to train a model on $\mathcal{T}_1$'s data, before fine-tuning it $\mathcal{T}_2$. This approach was used e.g. by Han and Eisenstein [16], who yield better results by task-tuning their model on CoNLL-2003 before fine-tuning it for historical NER. This approach, however, can only be considered in cases where $\mathcal{Y}_S = \mathcal{Y}_T$, or at least $\mathcal{Y}_S \cap \mathcal{Y}_T \neq \emptyset$. This requirement acts as a limit for tasks with very peculiar label spaces, as is the case in this study. Whereas generic tasks are provided with numerous labelled dataset that can be used as a first task-tuning step, specific tasks often lack human-annotated fine-tuning data. If generic entities such as persons or locations are frequent in open-access datasets, more specific entities like canonical citations are less furnished, constraining specific tasks to low-resource settings.

[1] See ConLL-2003 leaderboard for comparison : `https://paperswithcode.com/sota/named-entity-recognition-ner-on-conll-2003`.

To conclude this section is worth recalling a few takeaways. NER state-of-the-art models are attention-based and allow for efficient transfer learning strategies. Generic pre-trained models use language modelling as a source task and vast unlabelled corpora such as newspapers or Wikipedia articles as source data. They bear meaningful representations which can be leveraged to facilitate training on a target task. Transfer learning hence contributed to enhance state-of-the-art performances in a wide variety of target tasks and showed to be particularly helpful in domains where little annotated data is provided.

1.3 Goals

In the present study, we leverage transfer learning techniques to extract bibliographical entities from publications in Classics. Our entities are further detailed in Section 3 and include ancient authors and works, canonical references, reference scopes and references to fragments. Canonical references reflect the practice of referring to primary sources, which is frequent in Classical Studies. They can hint to an ancient author, to an ancient work or to a specific passage. For instance, "*[...] as mentioned by Thuc., III., especially 86-88...*" refers to the text spanning from the 86th to the 88th chapter of the third book of the History of the Peloponesian War by Thucydides. The example shows that such citations can be fragmented, incomplete, abbreviated and partly numeric. In that sense, canonical references certainly diverge from traditional entities. As such, their extraction can be construed as a domain- and task-specific application of NER. In such an environment, the extent to which transfer learning and attention-based models can enhance performances remains unknown.

Three transfer learning strategies can be applied to our case. The first consists in fine-tuning a model pre-trained on generic data, so that $\mathcal{D}_S = \mathcal{D}_{Generic}$. This method is the fastest as these models can be found off the shelf, but significant changes between $\mathcal{D}_S$ and $\mathcal{D}_T = \mathcal{D}_{Classics}$ may substantially curb the model's performances. The second strategy consists in fine-tuning a model entirely pre-trained on domain-specific data. This method may yield the best results as $\mathcal{D}_S \approx \mathcal{D}_T$, but it is computationally expensive and requires massive amounts of unlabelled domain-specific texts. To this day, no such model has been made publicly available for the domain of Classics. For reasons of time, this strategy was excluded from the scope of this study. The third strategy consists in fine-tuning a model pre-trained on both generic and domain-specific data, so that $\mathcal{D}_S = \{\mathcal{X}_{Generic} \cup \mathcal{X}_{Classics}, P_S(X)\}$, where $P_S(X)$ can be seen as a weighted average of $P_{Generic}$ and $P_{Classics}$. This can be done by continuing generic pre-training on domain-specific data. This method is less costly than the previous one. Furthermore, since $\mathcal{X}_T \subseteq \mathcal{X}_S$ and since $P_S(X)$ is partially dependant on $P_T(X)$, this method should smooth the difference between $\mathcal{D}_S$ and $\mathcal{D}_T$.

In this study, we compare several models and investigate the benefits of continuing pre-training on domain-specific data. We also test three fine-tuning strategies: baseline fine-tuning, frozen fine-tuning and fine-tuning with additional data. Finally, we evaluate the impact of data cleaning (see Section 5 for a detailed list of our research questions). In our case, continuing pre-training on domain-specific data does not yield significant benefits. The best results are obtained with the generic RoBERTa fine-tuned with additional data, which yields an F1 score of .82.

This report is organized as follows : Section 2 reviews related literature in domain-specific and task-specific NER. Section 3 presents the datasets and the entity typology used in this study. Section 5 presents the experimental design and the obtained results. Finally, Section 6 discusses theses results.

2 Related work

Two strands of comparable researches are examined below. The first deals with the adaptation of NER to domain-specific environments, the second with citation extraction. Both underpin the superiority of domain-specific pre-training and attention-based models.

2.1 Domain-specific NER

As shown by Augenstein et al. [17], NER systems based on static embeddings struggle to generalize to domain-specific corpora. For instance, using static embeddings, Riedl and Pado [18] compared CRF and bi-LSTM in generic and historical german data. A first notable finding is that domain-specific embeddings (fastText, Europeana) did not significantly take over generic embeddings (fastText, Wikipedia) when tested on historical data. Indeed, Europeana embeddings are respectively .01 above and .03 below Wikipedia embeddings' F1 score on the two historical datasets. This might be explained by the facts that their pre-training data is smaller and results of partially noisy optical character recognition (OCR) outputs. Another finding is that in the absence of a preliminary task-tuning phase, CRF performs better on the low-resource historical datasets. Authors conclude that CRF might be more flexible in a sparse environment. The question as to whether contextual embeddings and transformers can improve on these results is addressed by both Schweter et al. [19] and Labusch et al. [20]. Using the same datasets as [18], Schweter et al. show a general improvement using both static and contextual generic embeddings. Indeed, using three stacked layers of generic fastText embeddings trained respectively on Wikipedia, Common Crawl and Character-level, Schweter et al. obtain a new state of the art (+.03 in F1 score) on historical data. Their second experiment compares language-model-based system trained on different pre-training data. Results show that a meticulous choice of pre-training data (especially with an important overlap in time) is crucial. With a target domain data ranging from 1710 to 1873, pre-training data ranging from 1703 to 1875 yields the best language model, considerably surpassing the multilingual BERT. Labusch [20] exploits the transformer solution further, comparing the generic pre-trained multilingual BERT to a custom German BERT pre-trained only on domain-specific data. The latter tends to yield better results on historical datasets, though this improvement could also be attributed to it's purely German pre-training. Surprisingly, the model does not overcome Schweter's LSTM+CRF with mixed embeddings.

Research conducted in other specific domains also conveys uncertain conclusions regarding the efficiency of domain-specific pre-training. Using their domain-specific BioBERT for biomedical NER, Lee et al. [21] report a slight improvement of +.062 in F1 score. Their model is initialized with BERT and further pre-trained on PubMed abstracts and fulltexts (18B tokens). Another model is SciBERT [22], which is pre-trained on a broader scientific corpus containing 3.2B token from biomedical and computer science related articles. This time, the model is trained from randomly initialized weights and shows larger improvements on BERT, improving its results up to 6% in NER. Closer to the domain of classics, ArcheoBERTje [23] yields +.035 in F1 score when compared with the generic Dutch BERT. Like BioBERT, its pre-training continues the generic pre-training of BERT with domain-specific data (700M tokens). As a possible conclusion on domain-specific NER, in-domain pre-training seems generally to enhance performances, even though its improvement margin is not always striking and may be domain- or language-related. Besides, general conclusions remain hindered by the fact that authors seldom provide detailed information regarding training configuration, hyper-parameters, sentence segmentation or text pre-processing.

Table 1. Size and function of domain-specific datasets used in this study.

Dataset	Usage	Labelled	Tokens
JSTOR - *CS-raw*	pre-training	no	172M
JSTOR - *CS-cl*	pre-training	no	149M
JSTOR - *CSExt-cl*	pre-training	no	700M
EpiBau	fine-tuning	yes	1.13M

2.2 Task-specific NER: citation extraction

Romanello [2] used various shallow machine learning algorithms to extract canonical references from Classics publications. His investigations show that CRF perform significantly above Support Vector Machines and Maximum Entropy models. His CRF model is therefore kept as a baseline for this study and will be further detailed in Section 4.2. Working on comparable reference extraction tasks, Rodrigues Alves et al. [24] investigate the benefits of deep neural networks over shallow CRF. They show improvements ranging up to .06 in F1 score using LSTMs with both word and character embeddings. Their results also show a consistent benefit of pre-training static embeddings on domain-specific data.

3 Data

Terminology The data used in this study must be distinguished according to its usage and its domain specificity. First, pre-training data denotes the unlabelled corpora used to train language models. It can be generic ($\mathcal{D}_S = \mathcal{D}_{Generic}$) or domain-specific (in which case $\mathcal{D}_S \approx \mathcal{D}_T \approx \mathcal{D}_{Classics}$). As each of the generic models presented below uses its own pre-training corpora, generic pre-training corpora are listed in Section 4. Secondly, fine-tuning data denotes the labelled corpus used to train the target NER model. Its domain is $\mathcal{D}_T = \mathcal{D}_{Classics}$. A summary of domain-specific data is shown in Table 1.

Domain-specific pre-training data Domain-specific pre-training data is composed of publications issued by JSTOR, a digital library of academic journals, books, and primary sources. The data was obtained from JSTOR under the Data for Research (DfR) program[2] using JSTOR's dedicated platform, constellate[3]. Data has two classification systems: *Source Category* and *Text and data-mining (TDM) Category.* The former results of human annotations made at journal level. The latter results of an automatic classification performed at document level. To this date, it is only possible to request data from selected *TDM Categories.* However, as none of these categories expressly mentions classical studies, a broad query was passed[4].

The resulting dump contains humanities-related publications of various kinds (academic articles, news, chapters...) dating from the late 18th century to present. As many documents have been

[2] `https://about.jstor.org/whats-in-jstor/text-mining-support/`

[3] `https://constellate.org/builder/?start=1900&end=2021`

[4] The exact query is the following: All documents from JSTOR about Linguistics - Applied linguistics, Linguistics - Grammar, Linguistics - Language, Linguistics - Philosophy of language, Linguistics - Theoretical linguistics, History - Historical methodology, History - Historical periods, History - Philosophy of history, Arts - Art history, Philosophy - Applied philosophy, Philosophy - Axiology, Philosophy - Epistemology, Philosophy - Logic, Philosophy - Metaphilosophy, Philosophy - Metaphysics, Political science - Civics, Political science - Government, Political science - Military science, Political science - Political geography, Political science - Political sociology, Political science - Politics, Religion - Religious studies, Religion - Spiritual belief systems, Religion - Theology limited to document type(s) article, chapter, book from 1800 - 2021

Table 2. Set of entities used for the extraction of classical references in EpiBau Corpus.

Code	Description	Example
AAUTHOR	The name of an ancient author	**Sophocles**' Oedipus Rex was performed in...
AWORK	The name of a ancient work	The **Ajax** combines the epic legacy...
REFAUWORK	A formatted reference to primary text	The embassy mentioned in **Pliny, Nat. Hist.** ...
REFSCOPE	The scope of the reference	...compared it with Vergil, Georg., **IV, 149-218**
FRAGREF	A reference to a fragment	...was not as damaged as **fr. 6 WEST**

automatically recognized from scans, data is partially noisy. In order to create the training corpora, two selection steps are performed. The first selection step consists in keeping only documents written in English ($\approx 99\%$). The second selection step deals with a peculiar layout recognition error observed in few documents where narrow columns have been merged in a single paragraph. Resulting texts are considerably corrupted and excluded from all training corpora ($\approx 1\%$), as they would encourage the model to account spurious contextual relations.

For experimental purposes, JSTOR data is divided into two corpora: *CS*, which contains only publications counting "Classical Studies" as one of their source categories and *CSExt*, which contains *CS* and a random sample of documents picked within the source categories that matched most frequently with "Classical Studies". *CS* has a raw (*raw*) and a cleaned (*cl*) version. On the contrary to the raw version, the cleaned version consists of pre-processed texts. As these two conditions are used to measure the effect of noise on language modelling, pre-processing aims at trimming identified sources of noise. The most frequent source of noise results from the interposition of footnotes, captions or running headers within the main text of a document. Pre-processing therefore includes a step to remove recurrent tokens at the beginning and at the end of each page. Character misrecognitions constitute a second important source of noise. In the field of classical studies, these errors happen mainly with Greek characters or, presumably with poor quality scans. In order to avoid training on extremely noisy data, sentences with more than 50% punctuation or less than 40% words present in 600k words lexicon are discarded. The first criterion proves to be useful, as erroneously detected characters are often transcribed to punctuation marks by the OCR engine. Other pre-processing steps include accent-striping, web-links removal, numbers removal and de-hyphenation.

Fine-tuning and evaluation data The EpiBau Corpus[5] is composed of 4 annotated volumes of *Structures of Epic Poetry*, a compendium on the narrative patterns and structural elements in ancient epic. It is used both for fine-tuning and testing the models. The data contains 1.1 million tokens and 37500 annotated entities. Initially, the corpus served was made as a by-product of the semi-automatic creation of an index locorum for the publication [25]. The entity types labelled in the EpiBau Corpus are listed in Table 2. EpiBau is divided in train-, dev- and test-sets to a ratio of 70-15-15. Entity and token counts per split are displayed in Table 3. All the experiments reported below are run on Epibau v0.3.

4 Models

4.1 Transformers

As exposed in Section 1.1, transformers use an entirely attention-based encoder-decoder architecture. The transformers tested here slightly modify the architecture proposed by Vaswani et al. [13]

[5] `https://github.com/mromanello/EpibauCorpus`, private at the time of writing.

Table 3. Set of entities used for the extraction of classical reference in EpiBau Corpus.

	train-set	dev-set	test-set
Tokens	712462	125729	122324
AAUTHOR	4436	1368	1511
AWORK	3145	780	670
REFAUWORK	5102	988	1209
REFSCOPE	14768	3193	2847
FRAGREF	266	29	33
Total entities	13822	1415	2419

though, as they introduce bi-directional multi-head attention. Multi-head attention consists in using several randomly initialised attention cells instead of a single one. Attention-heads outputs are concatenated and normalized to form each block's output representation. Bi-directional attention implies that each attention head consists in two separate attention cells trained on ordinary sequences and on inverted sequences respectively. The decoder to add on top of this multi-head attention encoder depends on the target task $\mathcal{T}_T$. For sequence labelling tasks like NER, both CRF and softmax can be chosen as decoders. In this study, only softmax was implemented. Fine-tuning the target model mainly optimizes the decoder layers, but gradient descent also actualizes the encoder's parameters. In the experiments described in Section 5, three transformer-models (BERT, RoBERTa and DistilBERT) are pitted against each other and compared to Romanello's CRF [2].

BERT BERT (Bidirectional Encoder Representations from Transformers) was introduced by [1]. The $BASE$ model is a bi-directional multi-head attention transformer with 12x768 encoder layers, which sum to 110 million parameters. It uses a 30,000 token vocabulary with WordPiece embeddings [26], a mechanism that uses bite-pair encoding to separate tokens into frequent atomic chunks. For example, the word "actualisation" gets tokenized to "actual" and "##isation", where the two hashtags indicate that a chunk is directly attached to the previous one. BERT is pre-trained using masked language modelling and next sentence prediction. The former consists in training a model to guess masked words in an input sequence. The latter is self-explanatory and mainly serves question answering purposes. Pre-training data is composed of English Wikipedia and BooksCorpus [1]. At the time of publication, the two corpora reached 3,3 billion words for a total of 16GB of uncompressed text. In this study, the cased English version of BERT$_{BASE}$ is used.

RoBERTa RoBERTa was introduced by [14] as an improved version of BERT. It shares the same architecture with BERT, uses the same embeddings and the same pre-training method. It is, however, pre-trained on an even larger pre-training corpus, combining about 160GB of uncompressed text from Wikipedia, BooksCorpus, CC-News, OpenWebText and Stories [14]. In this study, RoBERTa$_{BASE}$ is used.

DistilBERT DistilBERT was introduced by [15] as a distilled, smaller and faster version of BERT$_{BASE}$. It has the same architecture with only 6 encoder layers and 66 million parameters. Like BERT, it is trained on English Wikipedia and BooksCorpus for masked language modelling. In this study, the cased version of the English DistilBERT is used.

The common properties of these three transformers allow for a controlled experimental design

with only two parameters: the size of the pre-training data (BERT vs RoBERTa) and the size of the encoder (BERT vs DistilBERT).

4.2 CRF Baseline

In order to evaluate the benefits of transfer learning, transformers are compared to a CRF baseline developed by Romanello [2]. CRF are statistical models which can consider nearby elements when classifying a given example. This specificity makes them particularly appropriate to process textual data. The model used here leverages a rich set of features to extract bibliographical entities from pre-processed text. Pre-processing steps include language detection, sentences segmentation and part-of-speech (POS) Tagging. Features are hand-selected and comprise linguistic features (e.g. POS tags), word-level features (e.g. punctuation or capitalization) and semantic features. Among these is the presence of the token within two dictionaries covering multiple languages and containing names and abbreviations of ancient authors and works respectively. Training and testing is done using the corresponding splits from EpiBau Corpus.

5 Experiments, results and discussion

As mentioned in Section 1.3, we test two transfer learning strategies (generic pre-training and continued pre-training) which we cross with three fine-tuning strategies (basic fine-tuning, frozen fine-tuning and fine-tuning with additional data). Experiments are listed in Table 4. Our research questions are the following:

1. **Cross-model comparing.** Which model achieves the best performances (*bsl* experiments)?
   (a) Do transformers improve on CRF baselines (CRF *bsl* versus transformers *bsl*)?
   (b) Among transformers, which model works best?
2. **Pre-training.** How does further domain-specific pre-training affects the results ?
   (a) What is the effect of the noise present domain-specific pre-training data (*CS-cl bsl* versus *CS-raw bsl*)?
   (b) What are the benefits of continuing pre-training on domain-specific data (Generic pre-training *bsl* vs Continued pre-training *bsl*)?
   (c) What is the effect of quantity and relevance of domain specific pre-training data (*CS bsl* versus *CSExt bsl*)?
3. **Fine-tuning.** How does the fine-tuning strategy affect the results ?
   (a) What is the effect of keeping the encoder frozen (*frz* versus *bsl*)?
   (b) What is the effect of additional fine-tuning data (*add* versus *bsl*)?

For clarity of presentation, the experiments are grouped by fine-tuning strategy. The results of pre-training strategies are presented *bsl* experiments. This section is organized as follows: Sections 5.1 and 5.2 first present implementation and pre-tests. Section 5.3 presents the baseline fine-tuning experiments. Both models and pre-training strategies are compared, addressing questions (1) and (2). Sections 5.4 and 5.5 respectively focus on the general effects of freezing and adding supplementary fine-tuning data, respectively addressing questions (3.a) and (3.b).

Table 4. Experimental Design, where *bsl*, *add* and *frz* respectively stand for *Baseline*, *Additional data* and *Freeze*. BC stands for BooksCorpus while R refers to the specific pre-training corpora used for RoBERTa, as listed in Section 4. Finally, *CS* stands classical studies, *CSExt* for classical studies extended and *cl* for cleaned (cf. Section 3.

	Pre-training strategy	Pre-training Data	Model	Exp.
1	-	-	CRF	bsl
2	-	-	CRF	add
3	Generic pre-training	Wiki, BC	BERT	bsl
4	Generic pre-training	Wiki, BC	BERT	frz
5	Generic pre-training	Wiki, BC	BERT	add
6	Generic pre-training	Wiki, BC	DistilBERT	bsl
7	Generic pre-training	Wiki, BC	DistilBERT	frz
8	Generic pre-training	Wiki, BC	DistilBERT	add
9	Generic pre-training	Wiki, BC, R	RoBERTa	bsl
10	Generic pre-training	Wiki, BC, R	RoBERTa	frz
11	Generic pre-training	Wiki, BC, R	RoBERTa	add
12	Continued pre-training	Wiki, BC, CS-cl	DistilBERT	bsl
13	Continued pre-training	Wiki, BC, CS-cl	DistilBERT	frz
14	Continued pre-training	Wiki, BC, CS-cl	DistilBERT	add
15	Continued pre-training	Wiki, BC, CS-cl	BERT	bsl
16	Continued pre-training	Wiki, BC, CS-cl	BERT	frz
17	Continued pre-training	Wiki, BC, CS-cl	BERT	add
18	Continued pre-training	Wiki, BC, CS-raw	DistilBERT	bsl
19	Continued pre-training	Wiki, BC, CS-raw	DistilBERT	frz
20	Continued pre-training	Wiki, BC, CS-raw	DistilBERT	add
21	Continued pre-training	Wiki, BC, CSExt-cl	DistilBERT	bsl
22	Continued pre-training	Wiki, BC, CSExt-cl	DistilBERT	frz
23	Continued pre-training	Wiki, BC, CSExt-cl	DistilBERT	add

5.1 Implementation and evaluation settings

Experiments are implemented using HuggingFace[6], a framework which provides a wide panel of pre-trained models and language processing tools. The detailed scripts used to perform experiments can be found in the dedicated Github repository[7]. In the series of experiments reported below, the performances of the models presented in Section 4 are evaluated on EpiBau test-set. Two frameworks are used for NER evaluation: Seqeval [27] and CLEF-HIPE-2020-scorer[8]. Seqeval can be used directly within HuggingFace. It computes entity-based precision, recall and F1 score for each entity class in a strict way: entities are marked as true positives only if all the tokens constituting the entity are correctly predicted. Overall metrics are computed using micro-averages. This method averages at the level of entities and not of classes, which is

[6] https://github.com/huggingface/
[7] https://github.com/AjaxMultiCommentary/ner_for_classics
[8] https://github.com/impresso/CLEF-HIPE-2020-scorer

recommended when classes are imbalanced, as is the case in this study (see Table 3). Seqeval is only used for evaluation during training.

The CLEF-HIPE-2020-scorer is an entity-based evaluation module for named entity recognition and linking. It is used to compute micro-averages both in a strict and fuzzy way, the latter marking entities as true positives if at least one of its constituting tokens is correctly predicted.

Each scorer has specific settings and may compute the results differently. Unless specified otherwise, all results are presented in the CLEF-HIPE strict evaluation method. The choice of the evaluation method is further discussed in Section 6.

5.2 Pre-testing

A first series of tests was performed to estimate the minimal number of training epochs requested to reach optimal results. Figure 1 shows the evolution of F1 score, precision and recall during a 15 epochs. As expected by [1], no metric significantly improves after around epoch four. Following experiments were run with seven epochs.

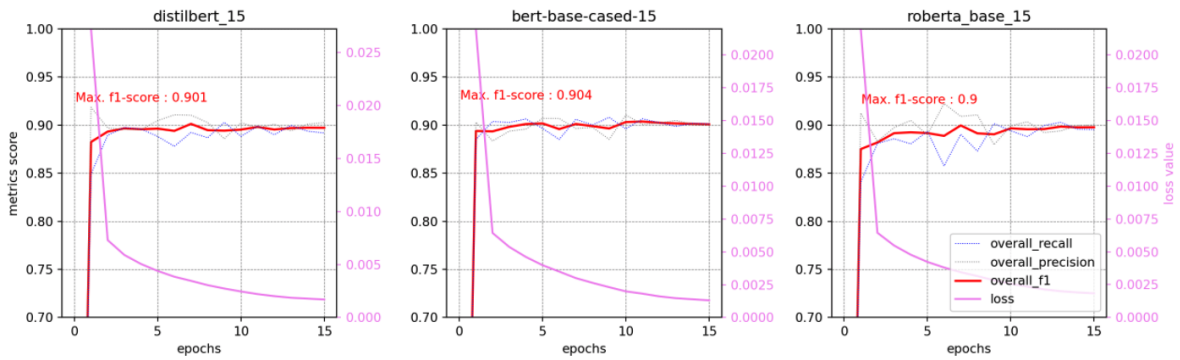

Figure 1. Evolution of metrics during fine-tuning. Evaluation is performed at the end of each epoch using seqeval.

5.3 Baseline fine-tuning

5.3.1 Comparing models

CRF vs Transformers We first address research question (1.a): Do attention-based models improve on CRF baselines? Our goal hereby is also to assess how the three generic transformers perform in the domain of Classics. BERT, DistilBERT and RoBERTa are therefore directly fine-tuned on EpiBau-train. In order to minimize the number of experiments, hyper-parameters are left to their default value[9]. According to general benchmarks, contextual representations and deeper neural architecture should allow transformers to take this round over. However, the domain-specificity of the task at hand reasonably influences these predictions.

HIPE-strict results of *bsl* experiments are shown in Table 5 and confirm these expectations, even though the gap between CRF and transformers is not as substantial as in comparable studies conducted on a more generic domain. Indeed, the generic versions of BERT, DistilBERT and RoBERTa improve the general F1 score "only" of 1.2%, 0.5% and 2.5% points respectively. In return, the CRF baseline remains higher in general precision, a superiority which can be explained by the use of gazetteers. These gazetteers however, cannot help capturing unregistered

[9] As set by HuggingFace's `TrainingArguments()`.

or altered entities, a limitation which can account for the lower recall of the CRF baseline. This superiority in precision holds true across all entity types and is particularly striking for ancient works (AWORK), where CRF's precision is about 7% higher than BERT's. However, with a recall score below 50%, the former can't match the latter's F1 score. Despite showing the best performances for REFAUWORK and REFSCOPE, the CRF baseline is held back by the poor recall scores in AAUTHOR and AWORK. It is also interesting to observe that CRF's results are less impacted by the evaluation method. As shown in Table 6, CRF gains 10% in general F1 score when evaluated in a fuzzy way, whereas transformers gain 12% on average. This indicates that more entities are incompletely captured by transformers (see Section 6).

The relatively small improvements yielded by transformers models must be discussed in the light of other areas of research. As mentioned in Section 1.1, transformers have been able to show notable improvements in generic tasks. In this domain-specific and data sparse environment, their true potential may still restrained.

BERT vs DistilBERT vs RoBERTa We now address research question (1.b) and pit the three transformers models against each other. As mentioned in Section 4.1, the chosen transformers allow for a controlled experimental design in which on only two parameters change: the size of the pre-training data (BERT vs RoBERTa) and the size of the encoder (BERT vs DistilBERT). Hypotheses are the following :

1. As a distilled, lighter version of BERT, DistilBERT should be faster to train but yield results inferior to BERT's.
2. As RoBERTa uses the same architecture than BERT with a larger pre-training data, it should be able to transfer more generic representation and therefore yield better results than BERT.

These hypotheses have been verified by several benchmarks and are confirmed once more in Table 5. With a general F1 score of .785%, BERT is approximately .07% above DistilBERT and 1.3% below RoBERTa. Though DistilBERT achieves a slightly better general precision than BERT, it is inferior in recall. Besides, if all three models are close competitors for REFAUWORK and REFSCOPE, the categories AAUTHOR and AWORK show greater differences in favor of RoBERTa. Besides, it may be noted that BERT's improvement over DistilBERT remains consistent in continued pre-training. With a heavier architecture, BERT is able to ingest more precise representations and yields slightly higher precision, recall and F1 score than DistilBERT. This being said, differences are extremely tiny. Regarding the difference in pre-training time, it may be more reasonable to use DistilBERT for quick experiments. For now, these experiments show that more robust representations obtained from larger pre-training corpora achieve slightly better results in a domain-specific area.

5.3.2 Comparing pre-training strategies

We now address research question (2): What are the benefits of continuing pre-training on domain-specific data? As it can be trained much faster than the two other transformers, DistilBERT is chosen as a baseline for pre-training experiments[10] and is further pre-trained on *CS-cl*, *CS-raw* and *CSExt-cl*. Continued pre-training is performed for 3 epochs with default parameters, on masked language modelling. It can be observed that each model's generic pre-training phase also

[10]Pre-training all the models would have been computationally costly and would have gone beyond the scope and time allocated to this research. Besides, in a first round of evaluation (performed before revision of the data), RoBERTa was not improving on BERT and DistilBERT. The model was therefore excluded from subsequent experiments.

Table 5. Strict CLEF-HIPE results for pre-training and fine-tuning experiments. For pre-training on Generic + CSExt-cl, (1) and (3) indicate the results of the model after 1 and 3 epochs of pre-training respectively.

			ALL			AAUTHOR			AWORK			FRAGREF			REFAUWORK			REFSCOPE		
			6274			1511			670			33			1213			2847		
Pre-training data	Model	Exp.	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R
-	CRF	bsl	.773	.844	.713	.63	.829	.508	.62	.856	.487	**.471**	**.667**	**.364**	.77	.782	.758	.868	.876	.86
		add	.8	**.858**	.749	.706	**.856**	.601	.659	**.866**	.531	.429	.522	**.364**	**.784**	**.796**	**.772**	**.879**	**.886**	.873
Gen	BERT	bsl	.785	.799	.772	.722	.78	.672	.718	.771	.672	.167	.154	.182	.731	.72	.742	.862	.856	.868
		frz	.773	.773	.774	.743	.778	.711	.722	.72	.724	.094	.077	.121	.704	.685	.725	.841	.834	.847
		add	.804	.811	.797	.775	.805	.747	.746	.799	.7	.135	.122	.152	.743	.728	.759	.867	.863	.87
	DistilBERT	bsl	.778	.802	.756	.719	.808	.648	.678	.773	.604	.145	.139	.152	.725	.707	.744	.858	.855	.862
		frz	.753	.768	.738	.693	.775	.626	.609	.668	.56	.137	.125	.152	.698	.673	.724	.844	.835	.853
		add	.798	.807	.789	.768	.81	.731	.724	.786	.67	.158	.14	.182	.717	.698	.737	.872	.868	.877
	RoBERTa	bsl	.798	.806	.79	.758	.801	.72	.777	.799	.755	.143	.135	.152	.742	.734	.751	.854	.85	.858
		frz	.779	.785	.773	.736	.787	.692	.751	.752	.751	.103	.089	.121	.726	.714	.739	.839	.835	.844
		add	**.821**	.827	**.815**	**.819**	.842	**.797**	**.796**	.818	**.776**	.149	.147	.152	.756	.755	.756	.863	.86	.866
Gen, CS-cl	BERT	bsl	.786	.801	.771	.747	.806	.696	.73	.812	.663	.132	.116	.152	.71	.697	.723	.858	.852	.865
		frz	.772	.783	.761	.73	.793	.676	.711	.762	.666	.162	.146	.182	.697	.675	.721	.846	.84	.851
		add	.808	.811	.805	.791	.818	.766	.768	.797	.74	.162	.146	.182	.735	.717	.753	.866	.861	.871
	DistilBERT	bsl	.783	.799	.768	.714	.788	.653	.719	.775	.67	.15	.128	.182	.728	.708	.749	.864	.86	.867
		frz	.761	.765	.758	.724	.779	.677	.688	.702	.675	.052	.045	.061	.701	.68	.722	.832	.821	.844
		add	.803	.813	.794	.771	.818	.729	.725	.772	.684	.133	.119	.152	.742	.722	.762	.873	.869	.876
Gen, CS-raw	DistilBERT	bsl	.786	.796	.776	.729	.78	.685	.715	.748	.685	.189	.171	.212	.723	.708	.739	.864	.861	.867
		frz	.773	.777	.768	.734	.793	.684	.669	.689	.651	.077	.067	.091	.722	.699	.747	.846	.835	.857
		add	.8	.808	.792	.766	.807	.729	.73	.77	.694	.135	.122	.152	.728	.713	.743	.873	.869	.877
Gen, CSExt-cl (1)	DistilBERT	bsl	.779	.799	.759	.71	.794	.641	.703	.776	.642	.139	.128	.152	.716	.701	.731	.863	.859	.867
		frz	.767	.771	.763	.736	.798	.683	.662	.674	.651	.088	.086	.091	.701	.672	.732	.842	.833	.852
		add	.798	.807	.789	.763	.808	.722	.716	.766	.673	.154	.156	.152	.721	.701	.742	.874	.868	**.879**
Gen, CSExt-cl (3)	DistilBERT	bsl	.78	.796	.764	.727	.793	.67	.673	.753	.609	.119	.118	.121	.723	.706	.741	.86	.853	.867
		frz	.76	.768	.752	.742	.81	.685	.634	.666	.606	.087	.083	.091	.68	.656	.705	.839	.829	.849
		add	.805	.808	.802	.801	.831	.773	.752	.784	.722	.085	.079	.091	.709	.689	.73	.87	.865	.875

Table 6. Fuzzy CLEF-HIPE results for pre-training and fine-tuning experiments. For pre-training on Generic and CSExt-cl, (1) and (3) indicate the results of the model after 1 and 3 epochs of pre-training respectively.

			ALL			AAUTHOR			AWORK			FRAGREF			REFAUWORK			REFSCOPE		
			6274			1511			670			33			1213			2847		
Pre-training data	Model	Exp.	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R
-	CRF	bsl	.879	**.960**	.81	.696	.917	.561	.632	.871	.496	**.549**	**.778**	.424	.942	.957	.928	.979	**.988**	.97
		add	.895	**.960**	.838	.76	**.921**	.647	.67	**.881**	.54	.464	.565	.394	**.949**	**.963**	.935	.98	**.988**	.973
Gen	BERT	bsl	.906	.922	.891	.814	.879	.758	.767	.824	.718	.25	.231	.273	.927	.913	.941	.982	.975	.989
		frz	.911	.911	.912	.84	.879	.804	.796	.794	.799	.212	.173	.273	.92	.895	.947	.981	.973	.988
		add	.919	.927	.911	.856	.889	.825	.783	.838	.734	.324	.293	.364	.93	.911	.95	**.984**	.98	.988
	DistilBERT	bsl	.901	.928	.875	.8	.899	.721	.739	.842	.658	.377	.361	.394	.912	.89	.937	**.984**	.98	.988
		frz	.888	.906	.871	.783	.876	.707	.713	.783	.655	.301	.275	.333	.903	.872	.937	.976	.966	.987
		add	.91	.921	.9	.843	.889	.802	.767	.834	.71	.342	.302	.394	.909	.885	.935	.983	.978	.987
	RoBERTa	bsl	.917	.926	.907	.849	.897	.806	.815	.839	.793	.314	.297	.333	.915	.905	.926	.982	.977	.986
		frz	.911	.918	.904	.834	.891	.783	.799	.8	.799	.256	.222	.303	.919	.903	.936	.981	.975	.986
		add	**.928**	.935	**.921**	**.889**	.913	**.865**	**.822**	.844	**.801**	.269	.265	.273	.921	.921	.922	.983	.98	.987
Gen, CS-cl	BERT	bsl	.91	.928	.893	.828	.893	.772	.763	.848	.693	.289	.256	.333	.927	.91	.944	.983	.975	**.990**
		frz	.906	.919	.893	.817	.888	.757	.759	.814	.71	.297	.268	.333	.92	.891	**.951**	.983	.976	.989
		add	.924	.927	**.921**	.874	.903	.846	.808	.839	.779	.432	.39	**.485**	.925	.903	.948	.983	.977	.988
	DistilBERT	bsl	.901	.919	.883	.791	.874	.723	.769	.829	.716	.3	.255	.364	.917	.892	.943	**.984**	.98	.987
		frz	.903	.907	.899	.816	.877	.763	.779	.795	.764	.312	.273	.364	.92	.894	.949	.974	.96	.987
		add	.912	.922	.901	.838	.889	.792	.776	.826	.731	.32	.286	.364	.918	.894	.943	.983	.979	.988
Gen, CS-raw	DistilBERT	bsl	.907	.919	.896	.813	.87	.764	.773	.808	.74	.405	.366	.455	.923	.903	.943	**.984**	.98	.988
		frz	.904	.909	.898	.82	.886	.764	.777	.799	.755	.231	.2	.273	.917	.887	.948	.977	.964	.989
		add	.913	.922	.904	.836	.881	.796	.78	.823	.742	.405	.366	.455	.924	.906	.943	.982	.978	.987
Gen, CSExt-cl (1)	DistilBERT	bsl	.9	.924	.877	.789	.884	.713	.747	.825	.682	.278	.256	.303	.923	.904	.943	**.984**	.979	.989
		frz	.899	.904	.895	.821	.89	.762	.747	.76	.734	.235	.229	.242	.91	.873	**.951**	.976	.965	.987
		add	.911	.921	.901	.84	.89	.795	.763	.815	.716	.338	.344	.333	.916	.891	.942	.983	.977	.989
Gen, CSExt-cl (3)	DistilBERT	bsl	.903	.922	.885	.809	.883	.747	.731	.817	.661	.418	.412	.424	.924	.903	.947	.982	.974	**.990**
		frz	.898	.907	.888	.825	.901	.761	.739	.775	.706	.29	.278	.303	.908	.876	.941	.971	.959	.983
		add	.92	.924	.917	.872	.905	.841	.799	.833	.767	.31	.289	.333	.916	.89	.943	.982	.977	.988

includes the task next sentence prediction. However, since this task mainly serves the benefits of question answering [1], it is not performed here. Besides, Han et al. [16] achieved promising results by further pre-training BERT for masked language modelling only. In order to have an element of cross-model comparison, we also trained BERT on *CS-cl*.

Cleaned versus raw data In order to conduct the following pre-training experiments with optimal data, we first address research question (2.a): What is the effect of the noise present in domain-specific pre-training data? To answer this question, we compare the results obtained by further pre-training DistilBERT on *CS-cl* and *CS-raw*. *CS-raw* is a 172M tokens corpus. When cleaned in the way described in Section 3, its size shrinks to 149M tokens. Pre-training is continued for 3 epochs, so that the language models sees a total of ≈400M tokens. As mentioned above, it should be noticed that the size of this additional pre-training is largely inferior to the size of the generic pre-training data DistilBERT has already seen (3.3B tokens). For this experiment, expectations are uncertain. Cleaning the data may feed the model with enhanced data, but raw data may as well improve the model's robustness.

Results are almost identical, with a feeble advantage for the raw model (+0.3% F1 score), a trend also observed in the freezing experiments. This result is difficult to interpret. Indeed, the bonus of pre-training on noisy data is eventually to increase the model's robustness to OCR imperfections. However, fine-tuning data is born digital and therefore not subject to this kind of noise.

Generic pre-training versus continued pre-training We are now addressing both research questions (2.b) and (2.c): What are the benefits of continuing pre-training on domain-specific data? What is the effect of quantity and relevance of domain specific pre-training data ? It is difficult to formulate clear-cut hypotheses for these experiments. Globally, further pre-training is expected to help the language model fitting the target domain better. However, related works show (see Section 2) its benefits to be irregular. Hypotheses pertaining the quantity of additional pre-training data also remain dubious. One can argue with Schweter et al. [19] that the selection of the pre-training data is crucial and that training only on classical studies related data would allow the language model to gain representations that are closest to the target domain. However, pre-training on a larger humanities-related dataset may also yield better results as the size of the pre-training data increases.

In the experiments reported here, continuing pre-training on domain-specific data yields no significantly superior results. When further pre-trained on *CS-cl*, DistilBERT only gains 0.5% F1 score. The improvement margin is even tighter for BERT. However, it is interesting to see that AAUTHOR and AWORK gain from pre-training on classical studies with BERT. However, this improvement is not really followed by DistilBERT, which hampers general conclusions.

With such a little improvement, one can be tempted to conclude that that additional pre-training data is not sufficient to produce significant improvement. Indeed, *CS-cl* contains only 149M tokens, which is maybe too small to produce significant improvements. Results, however, are not better with the larger *CSExt-Cl* (≈700M tokens) though. Table 5 shows no significant improvement in fine-tuning, neither after 1, nor after 3 epochs, even though language model's perplexity[11] drops from 7.62 to 6.83.

[11]Perplexity is an intrinsic metric commonly used to assess the performance of a language model. It is inversely proportional to the probability given by the model to the test-set of the pre-training corpus, which should be high if the model predicts the test data well. A drop in perplexity therefore means a better language model.

5.4 Fine-tuning frozen models

In order to measure its true contribution in fine-tuning, the encoder is partially frozen in this experiment. Freezing a layer prevents its parameters to be actualised by gradient descent, hitherto keeping the representations of the pre-trained model unchanged. Pre-testing runs showed that freezing the entire encoder yields catastrophically poor results. It was therefor chosen to freeze half of the encoder blocks: 6 layers for BERT and RoBERTa, 3 for DistilBERT.

Hypotheses for this experiment are based on the following lines of thought: if pre-trained representations fit the target domain well, freezing the model should produce results that are close to *bsl* experiments. On the contrary, if these representations need to be considerably updated during fine-tuning, freezing should yield inferior results. The following hypotheses can hence be formulated :

1. As freezing partially impedes the capacity of the model to adapt to the target domain, freezing experiments should yield results slightly inferior to baseline experiments.
2. As continuously pre-trained models should fit the target domain better, freezing should have less impact on continuously pre-trained models than on generic models.

Results are shown in Table 5 and allow to confirm the first hypothesis. In average, freezing the models yields an F1 score 1.7% inferior to baseline fine-tuning results. The second hypotheses is more delicate to confirm. For the generic DistilBERT, the difference in general F1 score between *bsl* and *frz* is of 2.6%. For the continuously pre-trained model, this difference is of 1.7% in average, which goes is the direction expected by the second hypothesis. For BERT however, the differences were of 1.4% and 1.2% respectively, which disproves the hypothesis. These tiny difference between differences is difficult to interpret, especially without cross-validation.

5.5 Fine-tuning on additional data

To measure the benefits of additional fine-tuning data, models are fine-tuned on both train- and dev-set in this experiment. The dev-set adds ca. 149k tokens and 5526 entities, augmenting the *bsl* training data by ±20%. This experiment is expected to yield results systematically superior to baseline fine-tuning. Results confirm this hypothesis and show consistent improvement between +1.9% and +2.5% in F1 score. This allows RoBERTa to reach the highest F1 score recorded in all experiments: 82.8%. It is interesting to see that this improvement is not distributed equally across classes. Indeed REFAUWORK and REFSCOPE seem to be captured quite rapidly by the models, which, regarding their structured morphology and the preponderance of numbers, is expected. Adding supplementary fine-tuning material only increases the results slightly in both these categories. However, AWORK and AAUTHOR gain significantly in F1 score, precision and recall, with an average of +3.5% and +5.5% in F1 score respectively. This improved also opens wider perspectives on task-tuning. Indeed, a profitable strategy for future works could be to augment fine-tuning with samples containing authors already annotated as "PERS" in generic data. Finally, one could have expected continuously pre-trained model to be less impacted by additional fine-tuning material, as they are supposed to need less adaptation to the target domain. This was not confirmed, as the average F1 score improvement is of 2% for both generic and continuously pre-trained models, with very little variance (±0.003).

6 General discussion

The experiments reported above call for a first series of observations pertaining the feeble impact of domain-specific pre-training and the evaluation method. They also convey a more detailed

error analysis, which is provided before general conclusive remarks.

Pre-training strategies With no significant improvement over the generic model, we cannot conclude that further pre-training the model helps in our case. This result is difficult to interpret. Even if domain-specific pre-training is supposed to enhance the source model's capacity to create representations that fit the target domain better, it is worth recalling that this technique does not always yield superior results. As mentioned in Section 2, other analogous publications show little or no improvement. BioBERT [21] for instance, yields an improvement +0.62% F1 score on biomedical NER despite being further pre-trained on 18B tokens of in domain-specific texts. Working with static embeddings, Riedl and Pado [18]) also enhance their results by a tiny margin with domain-specific pre-training. However, other models such as ArcheoBERTje [23] and Rodrigues Alves et al. [24] show significant improvements. In the experiments presented above, further pre-trained models were expected to yield superior results and potentially to be less impacted by freezing and by the addition of training material. None of these hypotheses proved to be true. Besides, no particular improvement was consistently observed in any entity type.

In such an uncertain context, it is not clear whether in-domain pre-training should be a priority for future works. It can be argued that in this study, continued pre-training has not been performed on sufficient data to create a real difference with generic pre-training, but that future researches could try to leverage more domain-specific pre-training data or to increase the number of epochs. One could also consider increasing the learning rate during language modelling. This would update generic representations more aggressively. However, the mixed results presented above tend to nuance this account.

Another possible explanation for the lack of improvement is that the domain-specific pre-training data used in this study does not exactly match with the target domain. *CS-raw* is a corpus of articles and chapters extracted from publications (mainly journals) manually classified as classical studies. As such, it should be closer to EpiBau in terms of features and distribution, but its quality is altered by recurrent OCR noise which is not present in the fine-tuning data. However, as an extensive cleaning of the data did not improve the results either, this explanation should be dismissed. This being said, further experiments should be conducted on noisy fine-tuning data in order to draw more stable conclusions.

Choosing the right evaluation method A critical point of this study lies in the choice of evaluation method. As mentioned in Section 5.1, three evaluation methods have been used : Seqeval, CLEF-HIPE-strict and CLEF-HIPE-fuzzy. In a first round of evaluation which determined the experimental design, models were evaluated using seqeval, which yielded results far above CLEF-HIPE-strict. This difference was difficult to explain as both tools are supposed to be entity-based and CoNLL compliant[12]. After several experiments, the error showed to be due to an implementation error in HuggingFace which this research led to fix[13]. On the other side, the use of the CLEF-HIPE-scorer also conveys several remarks. First, as CLEF-HIPE only receives IOB inputs, transformers' tokenized predictions must be reconstructed. As detailed in Section 4, BERT, DistilBERT and RoBERTa use their own WordPiece tokenizers, which separate tokens into smaller chunks. Reconstructing predictions implies to mark each token with the dominant label among the chunks composing it. This method however, can lead to conflicts when two chunks belonging to a same token are marked with different labels. With an average of 80 conflicts per model, reconstruction also accounts for the difference between CLEF-HIPE and

[12]See `https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt` for the original perl-script.
[13]See issue `https://github.com/huggingface/transformers/issues/14043`.

Seqeval. A second series of remarks must address the differences between the strict and fuzzy evaluation methods. It is first worth mentioning that these differences are extremely important. In average, fuzzy scores are 11% above strict scores in average. Besides, the differences observed between evaluation methods are not homogeneous between categories. The results presented in this study should therefore be handled with caution, which is furthermore confirmed by error analysis.

Error analysis We now take the results of the best model and proceed to an in-depth analysis of its confusion matrix. As shown in Table 5, the generic RoBERTa with additional data has the highest general F1 score. It is therefor chosen as a basis for error analysis. Within RoBERTa's confusion matrix, we first focus on ancient authors (AAUTHOR). In total, the model misses 239 occurrences of AAUTHOR. Counts show that christian poet *Arator* is most often missed with 21 false negatives for a total of 69 occurrences in the test set[14]. Even though no direct pattern could be identified, two remarks can be made. First, *Arator* is one of the "partial entities", which happen when a word is not tokenized properly in the original IOB document. Secondly, *Arator* only appears in one of the documents constituting EpiBau, in which the token is sometimes annotated as an AUTHOR, sometimes as a REFAUWORK. This mitigated result encourages a two-level annotation system in which an author ought to be nested in a formatted reference, instead of being annotated as REFSCOPE. This method will be preferred for future annotation guidelines and campaigns. After *Arator*, *Ovid* and *Appolonius* are most frequently missed, with 19 and 15 misses respectively. Despite careful analysis, no particular pattern cound be found to explain the case where the model failed. Analysing false positives also yields interesting results which can hint to forgotten annotations in the test-set. Unsurprisingly, the most common mistake lies in the inclusion of the possessive mark (e.g. "Aristotle**'s**") in an AAUTHOR entity. Even though annotation guideline specify that possessive should not be included, some erroneous 288 cases remain in the training data and lead the model to errors. The same holds true for commas directly following an AAUTHOR entity. In total, commas and possessives account for 40% of RoBERTa's false positives (82 of 201). It is striking to see that the remaining false positives are almost exclusively entities which have been forgotten in the ground-truth (*Lucretius*, *Homer*, *Ovid*...). Apart from two mythological characters annotated as AAUTHORS (*Anna Perenna* and *Byblis*), it seems that RoBERTa would have reached a precision score close to perfection on flawlessly annotated data. Without going in depth in all categories, it should be pointed that AWORK false positives also hint to missed annotations, as ancient works like the *Gospels*, *De rerum Natura* or the *Aeneid* are the most frequent sources of false positives. Finally, we analyse the most numerous categories: formatted references (REFAUWORK) and of reference scope (REFSCOPE). Here, the model seems to be mainly confused by punctuation marks such as opening and closing parentheses or brackets, commas and period. In average, these account for more than 60% of false negatives and about 93% of the false positives in these two categories. This result hints to potential inconsistencies in the annotation.

Error analysis is encouraging, as it shows that many errors are not due to the model, but to the data itself. Despite cautious annotation work and double checking, perfect data remains out of reach and cannot be a reasonable goal. However, it is important to know that the predictions of the best model are actually better than reported above and probably lend to more stable downstream NLP pipelines.

Miscellaneous remarks. Finally, several experimental choices can be criticized. First, continuous pre-training experiments are almost only performed with DistilBERT despite the fact

[14]Notice that these counts are token-based.

that BERT and RoBERTa yield superior results. As a matter of fact, experimental design was devised after a first round of experiments performed on EpiBau v0.1, where RoBERTa's results were not as high as BERT's and only slightly superior to DistilBERT's. The two former models being both longer to fine-tune and to pre-train, they were excluded from pre-training experiments. As error analysis later showed important misses in the annotations of EpiBau v0.1, a revision campaign was conducted and models were retrained for fine-tuning, but not for pre-training. As the second round of experiments on EpiBau v0.3 proved it to be better, RoBERTa should be further pre-trained in future experiments. Secondly, all transformers have been tested with a softmax output layer only. In order to be coherent with the CRF baseline and be able to judge the sheer impact of transformer based representation, a CRF should have been used as output layer in transformers. Apart from these points, the main challenge for future work will be to explain the feeble results of continuous pre-training. The option of gathering more domain-specific data can be considered, so as the elaboration of specialised languages models trained exclusively on classics.

General conclusion The main motivation for this research is to improve NER in a domain- and task-specific environment. It addresses three main questions : Which model achieves the best performances ? How does further domain-specific pre-training affects the results ? How does the fine-tuning strategy affect the results ? We first show that transformer-based models yield better results than CRF. Among transformer-based models, we show that the largest model with the largest pre-training data tends to perform best. In the present study, continuing pre-training on domain-specific data spikes no significant improvement over generic pre-training. Among the three tested fine-tuning strategies, fine-tuning with additional training data yields the best results, with a strict F1 score up to .82% for RoBERTa.

7 Bibliography

[1] Jacob Devlin et al. 'BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding'. In: *arXiv:1810.04805 [cs]* (May 2019). arXiv: 1810.04805 [cs].

[2] Matteo Romanello. *(Ch. 4) from Index Locorum to Citation Network: An Approach to the Automatic Extraction of Canonical References and Its Applications to the Study of Classical Texts*. Mar. 2017. DOI: 10.5281/zenodo.439122.

[3] Erik F. Tjong Kim Sang and Fien De Meulder. 'Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition'. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 -*. Vol. 4. Edmonton, Canada: Association for Computational Linguistics, 2003, pp. 142–147. DOI: 10.3115/1119176.1119195.

[4] Leon Derczynski et al. 'Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition'. In: *Proceedings of the 3rd Workshop on Noisy User-Generated Text*. Copenhagen, Denmark: Association for Computational Linguistics, 2017, pp. 140–147. DOI: 10.18653/v1/W17-4418.

[5] Martin Krallinger et al. 'The CHEMDNER Corpus of Chemicals and Drugs and Its Annotation Principles'. In: *Journal of Cheminformatics* 7.S1 (Dec. 2015), S2. ISSN: 1758-2946. DOI: 10.1186/1758-2946-7-S1-S2.

[6] Özlem Uzuner et al. '2010 I2b2/VA Challenge on Concepts, Assertions, and Relations in Clinical Text'. In: *Journal of the American Medical Informatics Association* 18.5 (Sept. 2011), pp. 552–556. ISSN: 1527-974X, 1067-5027. DOI: 10.1136/amiajnl-2011-000203.

[7] Sinno Jialin Pan and Qiang Yang. 'A Survey on Transfer Learning'. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (Oct. 2010), pp. 1345–1359. ISSN: 1041-4347. DOI: 10.1109/TKDE.2009.191.

[8] J.-B. Michel et al. 'Quantitative Analysis of Culture Using Millions of Digitized Books'. In: *Science* 331.6014 (Jan. 2011), pp. 176–182. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1199644.

[9] Tomas Mikolov et al. 'Efficient Estimation of Word Representations in Vector Space'. In: *arXiv:1301.3781 [cs]* (Sept. 2013). arXiv: 1301.3781 [cs].

[10] Jeffrey Pennington, Richard Socher and Christopher Manning. 'Glove: Global Vectors for Word Representation'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.

[11] Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. 'Neural Machine Translation by Jointly Learning to Align and Translate'. In: *arXiv:1409.0473 [cs, stat]* (Sept. 2014). arXiv: 1409.0473 [cs, stat].

[12] Ilya Sutskever, Oriol Vinyals and Quoc V. Le. 'Sequence to Sequence Learning with Neural Networks'. In: *arXiv:1409.3215 [cs]* (Dec. 2014). arXiv: 1409.3215 [cs].

[13] Ashish Vaswani et al. 'Attention Is All You Need'. In: *arXiv:1706.03762 [cs]* (Dec. 2017). arXiv: 1706.03762 [cs].

[14] Yinhan Liu et al. 'RoBERTa: A Robustly Optimized BERT Pretraining Approach'. In: *arXiv:1907.11692 [cs]* (July 2019). arXiv: 1907.11692 [cs].

[15] Victor Sanh et al. 'DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter'. In: *arXiv:1910.01108 [cs]* (Feb. 2020). arXiv: 1910.01108 [cs].

[16] Xiaochuang Han and Jacob Eisenstein. 'Unsupervised Domain Adaptation of Contextualized Embeddings for Sequence Labeling'. 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). Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4238–4248. DOI: 10.18653/v1/D19-1433.

[17] Isabelle Augenstein, Leon Derczynski and Kalina Bontcheva. 'Generalisation in Named Entity Recognition: A Quantitative Analysis'. In: *Computer Speech & Language* 44 (July 2017), pp. 61–83. ISSN: 0885-2308. DOI: 10.1016/j.csl.2017.01.012.

[18] Martin Riedl and Sebastian Padó. 'A Named Entity Recognition Shootout for German'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 120–125. DOI: 10.18653/v1/P18-2020.

[19] Stefan Schweter and Johannes Baiter. 'Towards Robust Named Entity Recognition for Historic German'. In: *arXiv:1906.07592 [cs]* (June 2019). arXiv: 1906.07592 [cs].

[20] Kai Labusch, Clemens Neudecker and David Zellhöfer. 'BERT for Named Entity Recognition in Contemporary and Historic German'. In: *KONVENS*. 2019.

[21] Jinhyuk Lee et al. 'BioBERT: A Pre-Trained Biomedical Language Representation Model for Biomedical Text Mining'. In: *Bioinformatics* (Sept. 2019), btz682. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btz682. arXiv: 1901.08746.

[22] Iz Beltagy, Kyle Lo and Arman Cohan. 'SciBERT: A Pretrained Language Model for Scientific Text'. In: *arXiv:1903.10676 [cs]* (Sept. 2019). arXiv: 1903.10676 [cs].

[23] Alex Brandsen et al. 'Can BERT Dig It? – Named Entity Recognition for Information Retrieval in the Archaeology Domain'. In: *arXiv:2106.07742 [cs]* (June 2021). arXiv: 2106.07742 [cs].

[24] Danny Rodrigues Alves, Giovanni Colavizza and Frédéric Kaplan. 'Deep Reference Mining From Scholarly Literature in the Arts and Humanities'. In: *Frontiers in Research Metrics and Analytics* 3 (2018), p. 21. ISSN: 2504-0537. DOI: 10.3389/frma.2018.00021.

[25] Matteo Romanello. 'Experiments in Digital Publishing: Creating a Digital Compendium'. In: *Structures of Epic Poetry* BOOK_CHAP (Dec. 2019). DOI: 10.1515/9783110492590-074.

[26] Yonghui Wu et al. 'Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation'. In: *arXiv:1609.08144 [cs]* (Oct. 2016). arXiv: 1609.08144 [cs].

[27] Hiroki Nakayama. *Seqeval: A Python Framework for Sequence Labeling Evaluation*. 2018.