Master Thesis:
Developing a
Cross-Lingual
Named Entity Recognition Model

Jowita Podolak[1], Philine Zeinert[2]
[1]jopo@itu.dk
[2]phze@itu.dk

May 26, 2020

Abstract

To build a Cross-Lingual NER, we first need to transfer successfully between different languages. Hence, we need a language-agnostic model - one not learning and incorporating in its weights any language-specific features. This paper explores linguistic features' impact on multilingual Transformer models, considering Named Entity Recognition task, and with the assumption that those models are not, truly, language-agnostic. First, we describe how cross-lingual transfer is implemented, with the focus on state-of-the-art Transformer models. Then, in the experimental part, we leverage vast Wikiann datasets of Indo-European languages to fine-tune mBert and XLM-RoBERTa on NER task. By finding four main learnt features derived from European Sprachbund, we point out their cross-lingual weaknesses, that should be improved in the future.

Contents

1 Introduction

1.1 Motivation

Almost half of the homepages of the most visited websites on the Internet are in a language other than English.[**?**] Providing seamless experience in everyone's preferred language would serve as a significant improvement in the life of communities around the world. The advantage lies not only in social media platforms by offering features like recommendations, suggestions or policy-violating content, but also in much more significant issues.

Language is information. The variety of languages in the world is its beauty but also its curse - if we do not understand each other, we cannot communicate, and, as a result, we cannot solve problems that need our collaboration. And those tend to be most urgent - global warming, refugee crisis, pandemics.

Unfortunately, language is usually not seen as a priority in emergency responses. As a result, misinformation, mistrust, fear and panic can spread quickly. On the other hand, the information in languages people can understand can help save lives in a crisis; for instance, the language barrier was one of the main difficulties faced by humanitarian workers responding to the Ebola crisis in 2014.[**?**]

Cross-lingual systems are critical cause they help to break language barriers. While writing this thesis, we are living under an uncertain time of Coronavirus pandemic. To stop the disease from spreading or avoid another one coming, governments are proposing dangerous levels of surveillance. But people can comply without those extreme measures if they have trust in science, public authorities and media.[**?**] There is much more in establishing trust and improving cooperation between nations than just a good system for translation, a 'common language'. Nevertheless, it will get us a few steps closer to global unity.

In research of cross-lingual systems, we focused on Named Entity Recognition (NER) since named entities are parts of the written documents that carry the most semantic information. In the world of data abundance, it is crucial to derive little units with most meaning[7]. Moreover, those systems are highly dependent on a large amount of annotated data, which are scarce or even completely unavailable. Developing sophisticated NER models for each language is comparable to building a brand-new application and solving the problem from scratch (including the cost-intense creation of training data)[1]. Therefore, developing cross-linguality is crucial in this field of NER[15].

The goal is to apply existing NER models to different languages, especially with low-resources. Our way of achieving cross-linguality would be to develop systems that are language-independent. Current state-of-art models show significant difficulties to project learned entity labels from one language to another even if entities seem to have very similar words across different languages.

[1] https://engineering.fb.com/ml-applications/under-the-hood-multilingual-embeddings/

1.2 Structure of the Thesis

In our work, we explore and discuss existing architectures for cross-lingual NER and identify transformer models as the state-of-the-art in this field. Following, we investigate their ability to project entity labels from one language to another language by looking at previous research and, based on them, run our own experiments.

In the experimental setup, we introduce Wikiann annotated NER dataset and give an insight into our chosen language subset and linguistic properties. Further, we explain our experimental design for testing the validation of our hypotheses defined in the beginning of this chapter.

Finally, "Results and Analysis" gives an insight into our findings and the analysis of them. The whole work ends with Conclusions: coming back to out initial research question and ideas about future work.

1.3 Research Question

Our thesis addresses the question:

How language-dependent are multilingual transformer models for cross-lingual NER?

The objective of our research is to investigate experimentally how multilingual transformer models perform in zero-shot settings for cross-lingual Named Entity labeling. Cross-lingual results are still not competitive with monolingual results, which gives a strong indicator for existing language dependencies. We focus in our work on languages from the Indo-European area providing rich resources of annotated NER datasets allowing proper evaluation of our results.

Furthermore, our initial research let us formulate four main hypotheses where our experiments were tested against. They can be found in the chapter "Experimental setup".

2 Named Entity Recognition

Almost every text document contains particular terms which represent specific entities that are more informative and have a unique context. These entities are known as named entities - terms that represent real-world objects like people, places, and organisations.

Automatically detecting named entities in text and classifying them into predefined entity types is a fundamental information extraction task. Named entity recognition provides essential input for many applications, including relation extraction, entity linking, question answering and text mining. Building a fast NER model is a crucial step towards enabling large-scale automated information extraction and knowledge discovery on the huge volumes of electronic documents we deal with today [9].

2.1 Traditional techniques for building NER systems

In the past, NER strongly relied on domain-specific knowledge, with linguistic grammar-based techniques in rule-based systems or in the feature engineering. Expert knowledge, however, can be discarded completely by using big datasets and deep learning. Sometimes, the combinations of those two is used.

2.1.1 Rule-Based systems and Feature Engineering or Experts knowledge systems or Expert systems

Rule-based NER systems rely on hand-crafted rules, designed by experts. Due to domain-specificity and incomplete dictionaries, high precision and low recall can be observed from such systems [**?**].

In feature engineering, given annotated data samples, features are carefully designed to represent each training example. Machine learning algorithms, using standard feature-based classification approaches such as Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs), are then utilized to learn a model. Feature vector representation is an abstraction over text where a word is represented by one or many Boolean, numeric, or nominal values. Word-level features (e.g., case, morphology, and part-of-speech tag), list lookup features (e.g., Wikipedia gazetteer and DBpedia gazetteer), and document and corpus features (e.g., local syntax and multiple occurrences) have been widely used in various supervised NER systems. [**?**]

Systems based on experts knowledge were in demand even after the emergence of deep neural networks because of the scarcity of training instances - not only there is little training data but also named entities rarely appear within sentences. The drawback of rule-based NERs and feature engineering is that their computation and design are very time-consuming and those systems cannot be easily transferred to other domains. [**?**]

2.1.2 Deep Learning

Deep neural network architectures can release researchers from time-consuming human feature engineering by learning automatically and discovering hidden features. Those classification models are trained by showing a neural network large amounts of data labelled with named entities as examples. Through this process, they learn how to categorise new examples, and then can be used to make predictions for unseen sentences. Capturing long-term dependencies is essential in this task as many words within a sentence can influence each other, not only the closest neighbours. There are two popular deep architectures that are able to capture those long-term dependencies in a word sequence: the Convolutional Neural Networks (CNN) and the Recurrent Neural Networks (RNN). CNN considers the current input while RNN considers the current input and also the previously received inputs. RNN and its improved version - LSTM (Long Short Term Memory Networks) allows information to persist. Lastly, by concatenating left and right context representation (forward LSTM and backward LSTM consecutively) representation of a word in context is further improved[2]. This forward and backward LSTM is referred to as bidirectional LSTM. CNN and RNN developed using only the word embeddings achieved similar results as the state-of-the-art CRFs with human-engineered features and knowledge from dictionaries.[14]

There is no doubt that the introduction of neural architectures has significantly advanced NER. However, the success of these methods is highly dependent on a large amount of annotated training data. Thus it remains a challenge to apply these models to languages with limited amounts of labelled data.

2.2 NER for resource-poor languages

In order to build NER model for low- and medium-resource languages, we either need to 1) annotate data or 2) develop a model that does not need it but uses another dataset in service to the cause.

The first solution can be done with the Annotation Projection technique: generating training data by projecting labels from a monolingual data set in the source language (resource-rich) directly to the targeted language(resource-poor).

On the other hand, in the Direct Model Transfer technique (solution 2), a model that is trained in one or more languages is directly applied to the target language [8]. The idea is to develop universal features, that the model can apply to unseen languages [9]. Of course, combining the two methods (Annotation Projection and DMT) is also possible.

Below we discuss the achievements of both in detail.

[2]In section 4.1.1 we discuss limitations and improvements of those techniques

2.2.1 Annotation projection

Annotation projection depends highly on the quality of its parallel training corpora and the corresponding creation of word alignments. This method assumes that sentences in one language can be translated flawlessly to another language. In practice, it is not applicable to many cases. Another challenge for Annotation Projection is the unavailability of the translation services and cross-lingual resources. Not only there are not many parallel datasets, but also machine translation services are not available for resource-poor languages. Several approaches in the field of Annotation Projection deal with these problems in different ways, e.g., by using a lexicon and translation services (see work of **Mayhew et al. (2017)**), by combining dictionaries and bilingual word embeddings (**Xie et al. (2018)**) or by "Translate-Match-Projection" method (see more in **Jain et al. (2019)**). Each work depends on common characters between source and target language. Xie et al addresses the challenge of word order within the cross-lingual transfer with an added "self-attention" layer.

2.2.2 Direct Model Transfer

Direct model transfer learning is the idea of generating language-independent features for cross-lingual NER labelling. One known work within this field is **Tsai et al. (2016)**. Their approach is called "WikiME" (= Wikification using Multilingual Embeddings) and connects words to categories of Wikipedia entries. Monolingual word embeddings are trained by taking Wikipedia articles and creating embeddings for words, the title and titles of integrated hyperlinks referencing another Wikipedia entry. With small supervision, with the aid of a title embeddings, word embeddings are aligned between two languages. A benefit of this method is that it enables alignment also for languages for which no machine translation exists. Finally, giving the word in a target language to the NER model, possible English titles can be derived and the best-matching English title chosen based on contextual factors and their frequency in the corpus. The title allows then the entity labelling [12]. This system has the potential for different source languages or using a combination of them to train their NER model.

Tsai et al. show how shared features between languages can be created using Wikipedia as a resource. Other approaches to create features shared by several languages are made by **Bharadwaj et al. (2016)**, who uses the phonetic alphabet as a link between source languages and target language and **Täckström et al. (2012)**, who employ a large number of monolingual word clusters and aligns them with identifying consistent words over languages [3][13]. **Cottrel and Duh (2017)** creates a connection between closely related languages based on their character representation. It allows applying their neural CRF model on low-resource languages, which share a similar alphabet [5].

These Direct Model Transfer approaches still seem challenging to implement in a way that they can compete against the mentioned Annotation Projection solutions (which are, in

fact, monolingual). Nonetheless, these works step away from creating large training datasets and building models for each language. Instead, they offer a more efficient, cross-lingual approach.

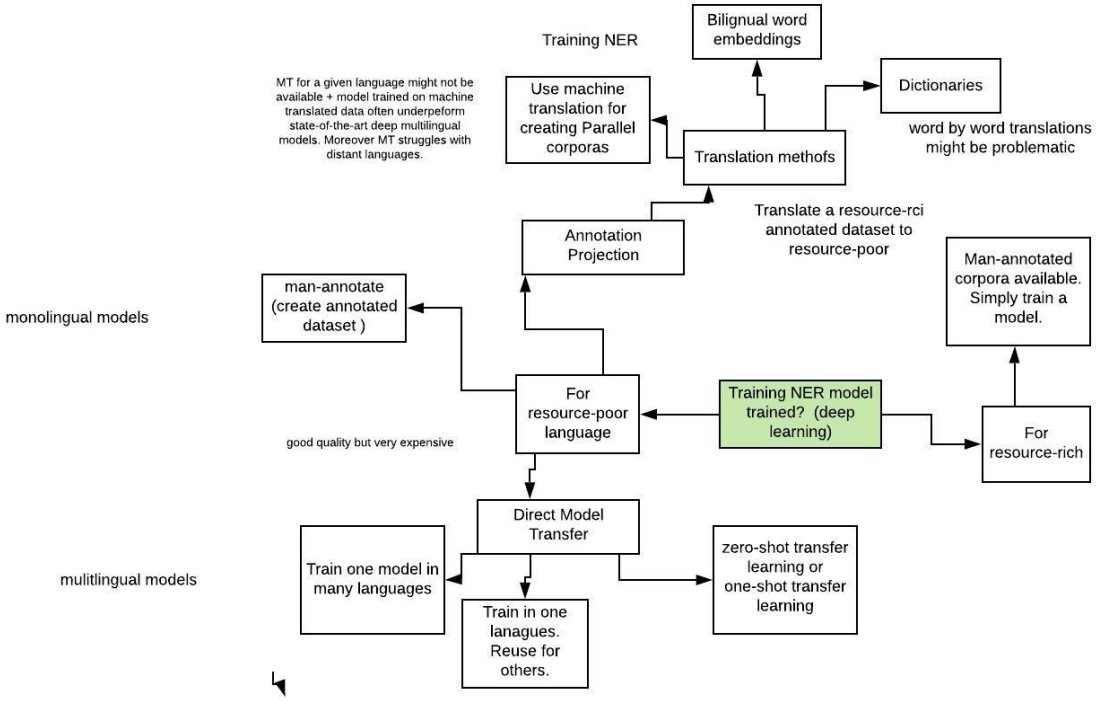

Figure 1: Training an NER model: monolingual and mulitlingual needs a better name

3 Cross-Lingual NLP

3.1 Word representations

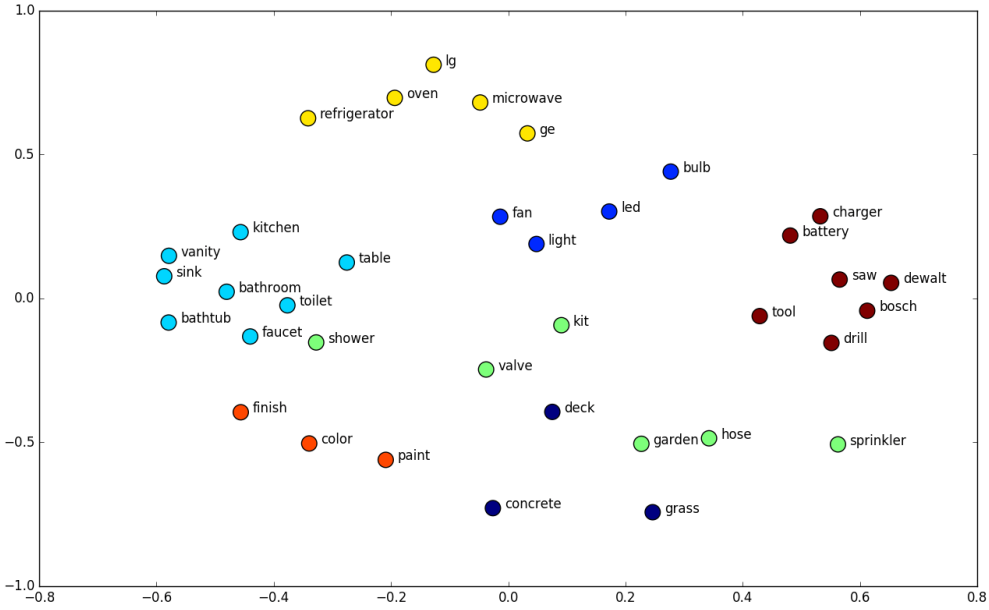

Figure 2: Words as vector representations

Source: (https://neptune.ai/blog/document-classification-small-datasets)

The choice of word representation, i.e., a transformation of raw text into a format understandable by machines - is the very first consideration in any NLP experiment. Simple word representations, like a dictionary (i.e., assigning ids to each word) or one-hot encoding (creating a vocabulary size vector filled with zeros except one), are not suitable for deep learning methods; dictionary may result in a model incorrectly assuming the existence of natural ordering and one-hot encoding has huge and sparse vectors that require much memory. Moreover, none of those techniques has a notion of similarity between words. However, it changed with the work of Mikolov et al. (2013) - Efficient Estimation of Word Representations in Vector Space - and their word2vec package. It introduced distributed word representations which are not only compact and dense (with each factor representing some distinct informative property) [**?**] but also capable of capturing similarities. For distributed word representation, if words in the space are close to one another, it means they

are similar to one another. The methods used for finding out those word embeddings are unsupervised. There are two types of distributed word representations: static and dynamic (contextualised).

3.1.1 Static Word Embeddings

We describe this group of embeddings as 'static' because the same word will always have the same representation regardless of the context in which it occurs. There are three most popular 'classical' word embeddings: the aforementioned Word2Vec, the GloVe, and the FastText (see Figure 4). Each has certain flaws. Word2Vec doesn't fully exploit global statistical information, and, just as Glove, fails to provide any representation for words that are not in the dictionary (i.e., in training). FastText, on the other hand, works well for words that were not supplied during training because of subword information. Instead of learning each word directly, it represents each word as an n-gram of characters. It allows embeddings to capture the meaning of short words and to understand prefixes and suffixes. [4] However, like all Static Word Embeddings, it does not capture polysemy. It means that static word embeddings work simply as lookup dictionaries - one vector belongs to each word, no matter the context. Therefore, if we feed two sentences:

Would you like a *cup* of tea?

The 2020 Women's World *Cup* final was hold in Australia.

Cup will have the same vector in both sentences. However, seeing this example, intuitively we feel we lose much meaning without considering the context. It is for this reason that traditional word embeddings fall short. They only have one representation per word; therefore, they cannot capture how the meaning of each word can change based on the surrounding context.

3.1.2 Dynamic Word Embeddings

The remedy for those problems is an idea of dynamic (contextualised) word embeddings, where the word vector vary for different sentences. Rather than pre-trained word embeddings we use a pre-trained model which assesses each time it sees a sentence, context of the words contained in it. Contextualised word embeddings address the issue of polysemous nature of words and thus capturing an even more meaningful representation of words.[3]

The backbone of dynamic word embeddings is language models. The core objective of language modelling is language understanding through hierarchical representations. Thus,

[3]https://towardsdatascience.com/from-pre-trained-word-embeddings-to-pre-trained-language-models-focus-on-bert-343815627598

the language model contains both low-level features (word representations) and high-level features (semantic meaning) whereas static word embeddings only contain low-level features.

Traditionally, a language model is a probabilistic model which assigns a probability value to a sentence or a sequence of words by always aiming to predict the next word given a previous sequence of words (mainly just predicting words in a blank - look at Figure 3). It can do this because language models are typically trained on massive datasets in an unsupervised manner. [**?**]

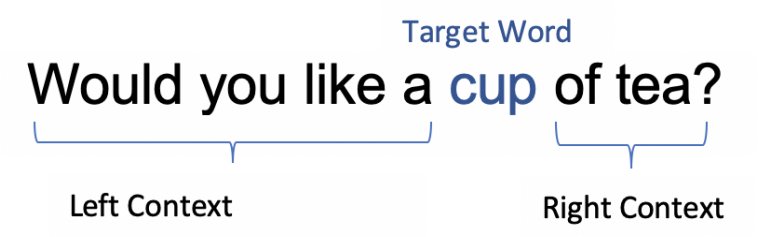

Figure 3: Generative aspect of Language modelling

One limitation of dynamic embeddings is that they cannot be used without a sentence-level context. [**?**] For example, dynamic embeddings cannot be directly used to solve lexical-semantic tasks such as word similarity or analogy. [**?**]

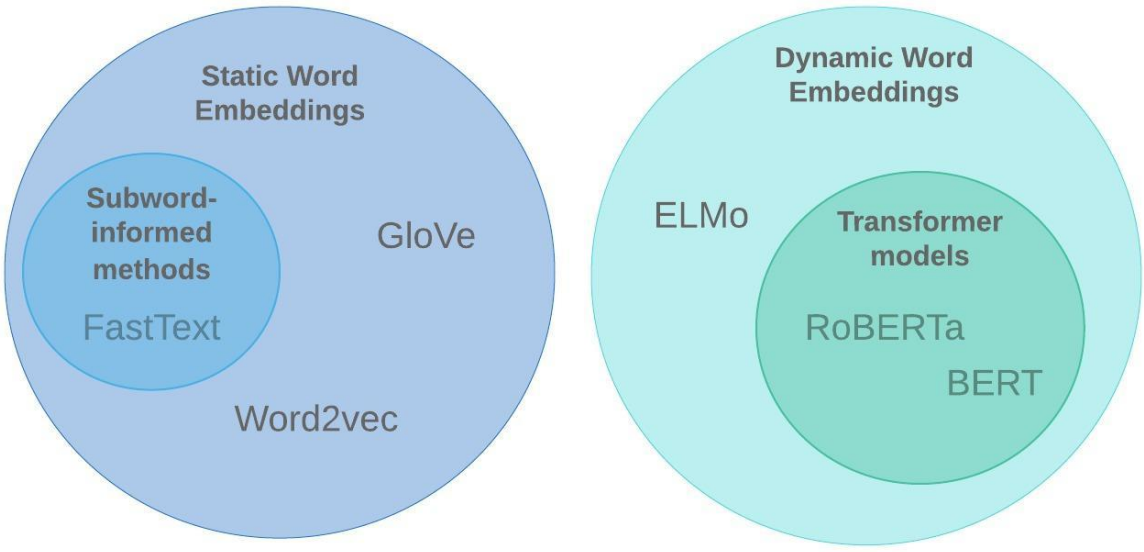

Figure 4: Word embeddings

3.2 Cross-lingual learning

Cross-lingual word embeddings transfer knowledge across different languages by representing words in a joint embedding space. In recent years, researchers proposed various models for learning cross-lingual representations. In the following, we divide them into two kinds - one that originated from static word embeddings - Aligned Word Embeddings, and one that essentially is a language model but trained with data in more than one language - Cross-lingual Language Model (see in the Figure 5 below).

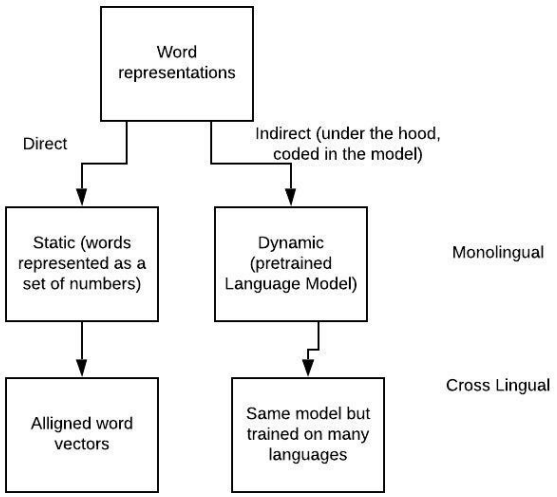

Figure 5: Monolingual and Cross-Lingual word representations

3.2.1 Aligned Word Embeddings and Cross-lingual Language Models training

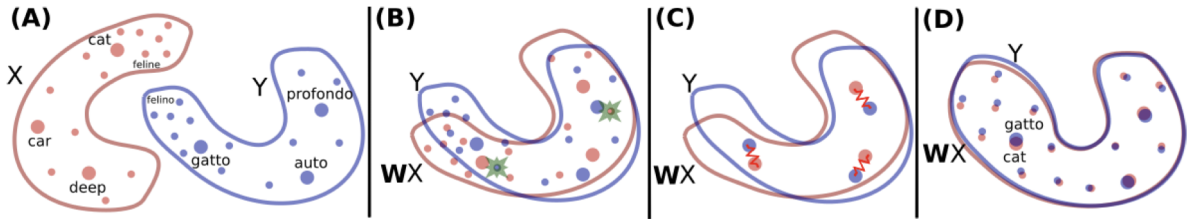

Figure 6: Aligning word embeddings

Source: [**?**]

Training aligned word embeddings can get quite complicated. The most straightforward approach is a supervised one (see supervised learning in the overview in Figure 7), e.g., using parallel corpora (sentences in one language versus the same sentence in the second language). However, it is costly to generate, especially for resource-poor languages, for which there is no machine translation system in place. Contrarily, monolingual data (for unsupervised learning) is easily available and thus training monolingual embeddings is much easier (bottom part of Figure 7). A breakthrough was the idea of leveraging orthogonal Procrestus analysis, which examines rotation for two or more objects to superimpose (overlap) in space. Thanks to this idea, it is enough to generate word embeddings in each language and apply a function that learns from a small parallel dictionary - even less than 100 words, to align them [**?**] (see Weakly-Supervised learning in Figure 7). The third method, an unsupervised one, is done with adversarial training and refinement.

Both weakly-supervised and unsupervised methods are implemented and pretrained for example by Muse Python library for multilingual word embeddings (word embeddings files can be downloaded simply as text files).[**?**]. Unfortunately, their performance drops for languages that are not closely related. The same applies as well when the number of languages grows. [**?**] Moreover, those solutions suffer from 'hubness problem' - some word vectors tend to be nearest neighbours of lots of other words which degrades the accuracy of finding the 'correct' neighbour.[6]

Pre-trained models can learn from unsupervised sources which are 'available in the wild'[4] For example, Multilingual BERT(see Figure 8) was trained from monolingual corpora in many languages. It does not have any marker denoting the input language nor have an explicit mechanism to encourage translation-equivalent pairs to have similar representations [11]. The answer to a question of how it learns without parallel data and rotation function is still unclear. Researchers hypothesize that having word pieces used in all languages (numbers, URLs, etc) which have to be mapped to shared space, forces the co-occurring pieces also to be mapped to a shared space, thus spreading the effect to other word pieces until different languages are close to a shared space. [11]

[4]In recent years, online services such as Wikipedia, social networking, gaming, Internet fora has provided researchers with large volumes of data.

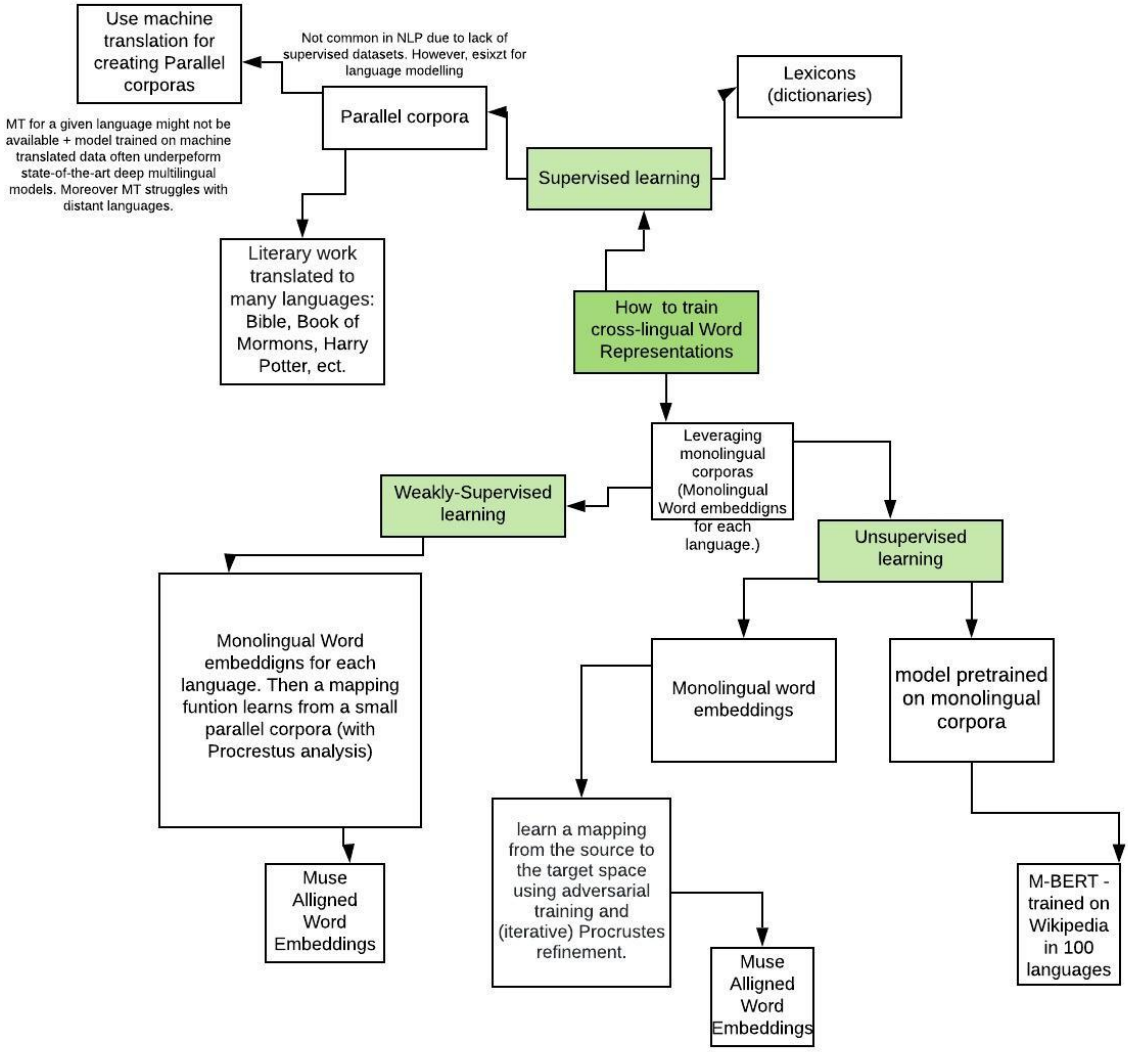

Figure 7: Training techniques of cross-lingual word representations

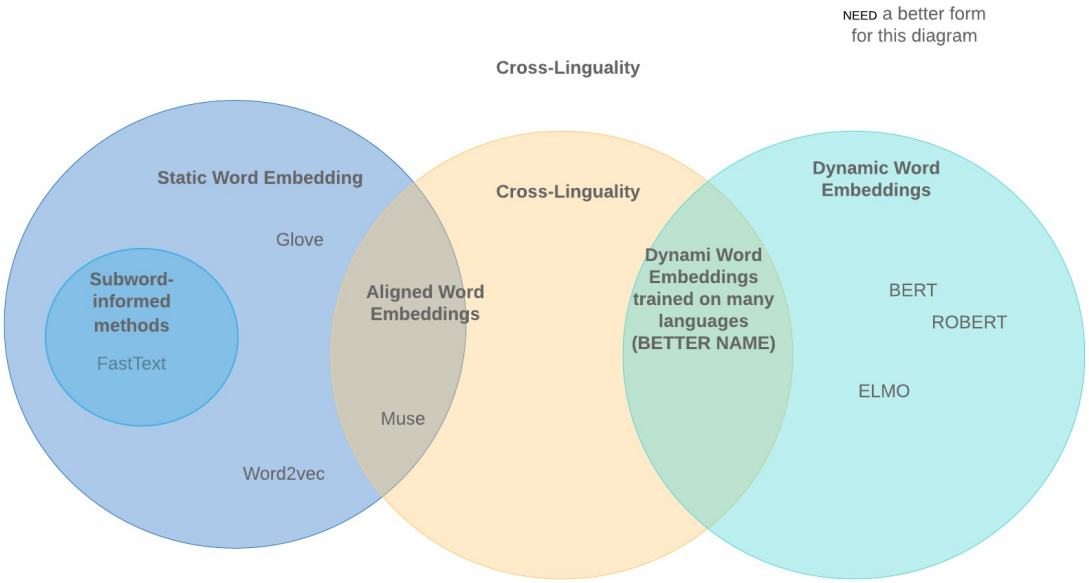

Figure 8: Cross Lingual systems

3.2.2 Cross-lingual versus multilingual models

Multilinguality removes the need for training multiple models for a collection of languages. A model which has access to multilingual information can learn generalizations across languages, in a similar way learning a new language can enhance the proficiency of a speaker's previous languages.

Cross-lingual embeddings attempt to map words that mean the same thing in different languages to almost the same vector. In contrast, multilingual embeddings do not guarantee any interaction between different languages (i.e. vectors are not intentionally overlapped). Aligned word embeddings are explicitly constructed to be cross-lingual (e.g., with rotation function mentioned before). Pretrained language models, on the other hand, are multilingual and indirectly cross-lingual, as it is likely that they may get somewhat aligned even if they were not directly designed for it.

3.2.3 Language-agnostic system

If technology developed for one language can be used for another merely by gathering data in the source language, then this system can be leveraged many times. It is advantageous for applications where collecting data requires much less effort than developing a model. architecture.[2]. For systems where training data is hard or even impossible to gather, we need a model that does not care about differences between languages - a language-agnostic system. If a system was able to predict labels for a language without being trained at all with this language (at least for the classification task at hand), we would offset dataset scarcity problem.

A truly language-independent system works equally well on all languages. If it is not, the system likely makes implicit assumptions about the language structure, i.e., it incorporates features specific to certain languages or group of languages. The more the model is language-agnostic, the higher is the overall performance of its direct model transferability. Moreover, it could teach us something about the nature of human language, and what human languages share in common. [2].

Systems are often wrongly deemed as 'language-independent', based solely on the fact that no linguistic knowledge was encoded in the training dataset. However, this does not ensure language independence. If language-independence cannot be claimed based on the architecture it has to, therefore, be proved explicitly - with experiments.

3.3 Transfer learning

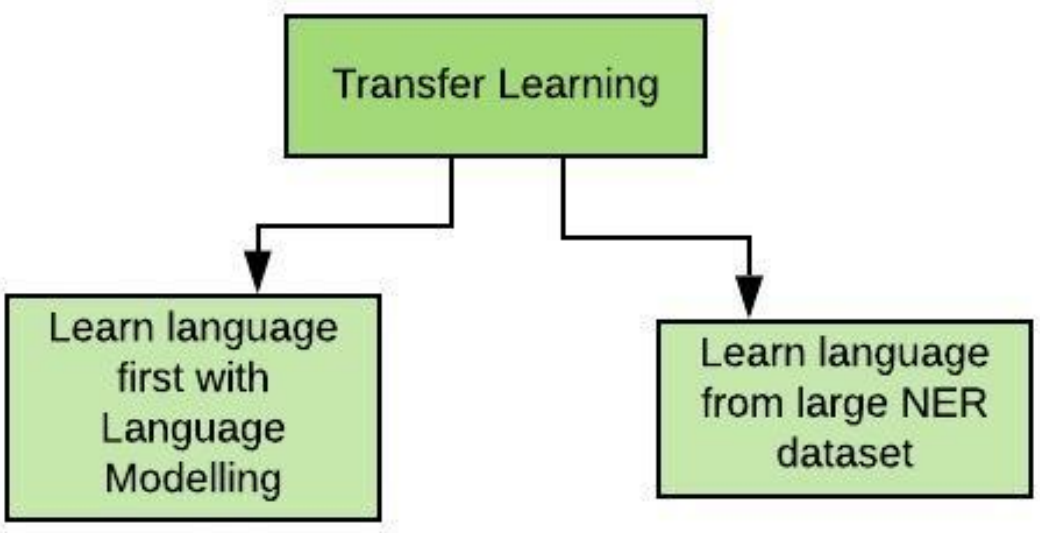

Figure 9: Transfer learning in NER. Left approach is more efficient - Language Modelling systems have much bigger datasets, available in the wild, and thus can learn language better.

Humans have an excellent ability to transfer knowledge across tasks. Learning new things is much easier due to similar (but not the same) tasks learnt in the past so that we rarely learn anything from scratch; for instance, knowing how to ice skate makes roller skating much easier.

In traditional machine learning, learning occurs purely on specific tasks and datasets, resulting in separate isolated models - we do not retain any knowledge which could be transferred from one model to another. Contrarily, the concept of transfer learning embodies learning abilities of humans. By reusing a pre-trained model (features, weights) for training new models, we not only overcome the problem of no or little data for classification tasks but also speed up the process of learning.

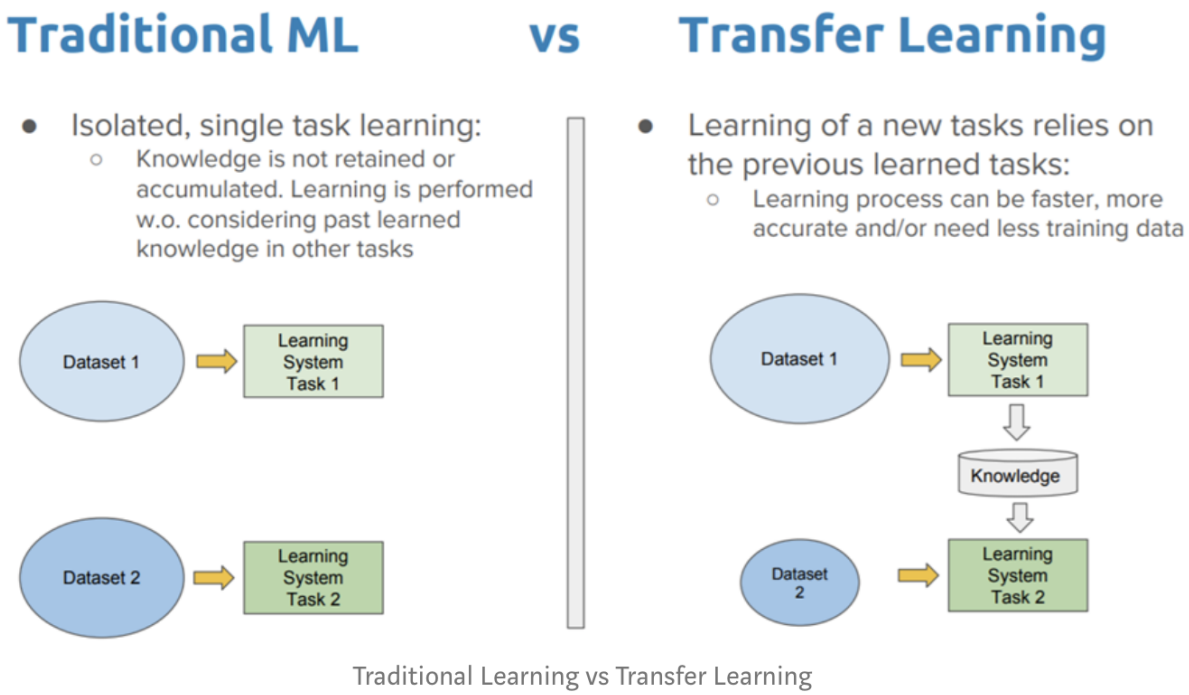

Figure 10: Traditional learning vs. Transfer learning

This concept is especially important in natural language processing: for every NLP task, language understanding is a foundation. Just like a child needs to first understand the language before learning to describe sights or tell stories, a model has to learn the language first, in order to perform further language tasks. However, learning a language requires much more data than what most annotated datasets for text classification can offer.

Since language has to be learnt by the text classification model, it is wiser to use systems that were built especially for that purpose, as a form of pretraining of the model. Provided that our dataset is not drastically different in context to the original dataset, the pre-trained model will already have learned features that are relevant to our task.

Earlier, we distinguished two ways in which to learn and represent a language - static word embeddings and dynamic word embeddings - pre-trained language models. In transfer learning, static word embeddings, pre-trained on large amounts of unlabeled data, are used to initialise the first layer of a neural network called embedding layer, the rest of the model is then trained on data of a particular task. This kind of transfer learning in NLP problems is shallow as learning is transferred to only the first layer of the model - the rest of the network still needs to be trained from scratch. To do so, NLP models initialised with these shallow representations still require big datasets to achieve good performance. [**?**]

With contextualised word embeddings, we can reuse all the layers, learning both low-level

features (word representations) and high-level features (semantic meaning). So, contextualised word embeddings not only capture usage variations across linguistic contexts (e.g., polysemy) but also include deeper knowledge about syntactic features of a language than static word embeddings do, making them a better candidate for transfer learning. In recent years, language-model-based contextualised word representation, such as ELMo , GPT, BERT, XLNet , and RoBERTa, has been proposed to replace static word embeddings as input representations, resulting in a massive leap in performance for many of the NLP tasks, such as text classification, natural language inference and question-answering.[**?**]

Another flavour to transfer learning in NLP, in addition to pre-trained language models, is utilising models trained on language-rich languages. Thus, the advantages of transfer learning can be leveraged twice in text classification.

3.3.1 Fine-tuning

Fine-tuning is a process based on the transfer learning concept. It means taking some model that has already learned something before (i.e. has been trained on some data) and retraining that model on a new data. In NLP transfer learning, we usually train on a general language modelling task and then fine-tune on the text classification task. Language model training is usually a long process, so researchers often download pre-trained models (we experimented with fastText, Glove, MUSE, mBERT and RoBERTa pre-trained word embeddings and models). Next, to enable classification, task-classification output layer - a tag decoder - is added. It takes context-dependent representations as input and produces a sequence of tags corresponding to the input sequence. Examples of architectures of tag decoders include a simple linear layer, softmax layer, conditional random fields (CRFs), and recurrent neural networks. Finally, we fine-tune by feeding new data into the obtained model.

3.4 Cross-lingual NER

In the Named Entity Recognition chapter, we discussed two ways of building models for resource-poor languages.

First, Annotation Projection entailed creating a model for each language (if in isolation, i.e., used without any additional methods). Alternatively, to build multilingual NER, we could implement an ensemble of NER models created either with human-annotated data sets or in combination with Annotation Projection.

The second approach was the Direct Model Transfer. Here, we use one single model which can later predict for languages it has not seen before in the training data (zero-shot transfer) or has seen very little data (one-shot transfer). This transfer can be significantly improved if we combine the idea of cross-lingual learning and transfer-learning, allowing the model to learn many languages first (with a pre-trained language model), and then learn NER tagging by fine-tuning on the annotated dataset.

Language model embeddings based on a novel architecture called Transformer, are becoming a new paradigm for NER. In the next chapter, we will explore Transformers, their architecture and pre-trained models, with the focus on multilingual models.

4 Transformer Models

4.1 Introduction

4.1.1 Delimitation of Transformers from RNN and CNN models

Transformers belong to the family of neural networks architectures and are used for every task where an input sequence is processed to a transformed output sequence (sequence transduction). [**?**] For this, it is necessary to have a memory that captures dependencies and connections between words within a sentence.[**?**]

One of the first architectures able to capture dependencies were RNNs[5]. However, because of the vanishing or exploding gradients problem, the model is biased by the most recent inputs and forgets older inputs. RNN's modified implementation that remembers for longer periods is a Long-Short Term Memory (LSTM) cells: it can prioritise information in the chain of nodes by being able to evaluate information and remember or forget them within each cell.[**?**] Nevertheless, LSTMs do not overcome the problem entirely and still forget words if sentences are long. Moreover, none of the two architectures is parallelisable, which is an important feature for large corpora.

Further improvement of RNNs' architecture is to include an Attention Decoder. Before, a hidden state was created for each sentence. Contrarily, in attention technique, every word is accompanied by a hidden state (with the information processed so far from the sequence from right to left and forwarded to the decoding). It assumes that every word can keep important information, and the network can pay different attention to each word in every decoding stage. Nonetheless, processing words in parallel is still not possible with this architecture.

The alternative to RNNs is Convolutional Neural Networks (CNN), which allow parallelism by processing words independently from each other in the input sequence. The execution time is much better in comparison to the approach of RNNs, but here just the local dependencies are stored along the way.

Transformers' edge lies in eliminating recurrence and convolution and replacing them with the self-attention. Thus, they are a combination of Convolutional Neural Networks (CNN) with the self-attention technique from the Recurrent Neural Networks (RNN).

4.1.2 The learning process in transformer models

Transformer architecture has an encoding stage and decoding stage. Each element is processed through the sequence of six encoders and decoders, each having a self-attention layer and a Feed-Forward-Neural-Network-Layer (see Figure 12). Encoding is done in parallel for each single word of the sequence (like in CNNs).

[5] see section 2.1.2

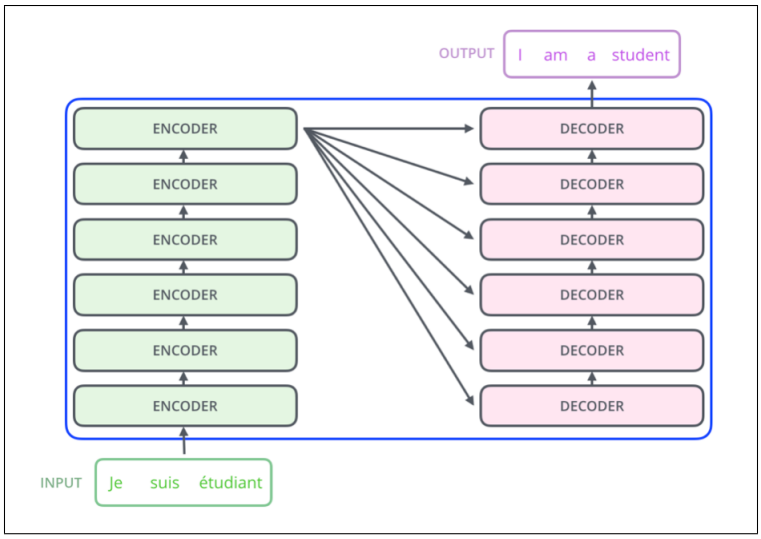

Figure 11: Encoding and decoding stage in Transformer models

Source: [**?**]

The downstream decoders also have an Encoder-Decoder-Attention layer (between the two aforementioned layers) computing the hidden states created for each word in the sequence, which allows in the decoding phase to also take into consideration the knowledge from other words in the sequence.[**?**]

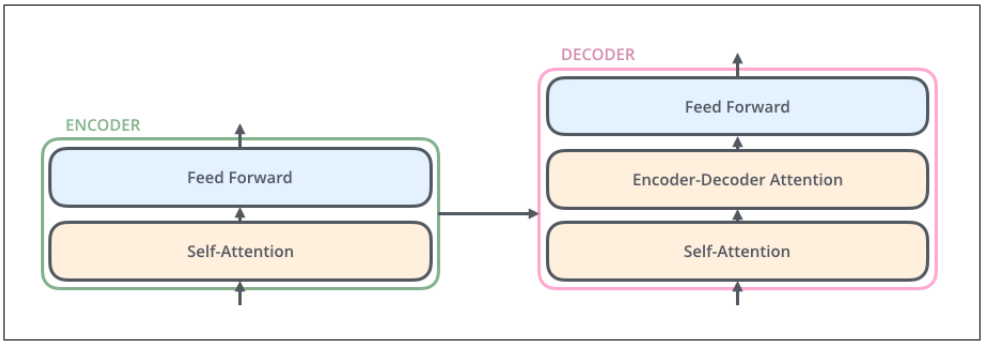

Figure 12: Encoders and decoders in Transformer models

In the self-attention layer itself, each word is computed together with a query, key and value vector (see Figure 13). All three are much smaller than word vectors (for space-efficiency) and can be thought of as metadata of the word. The initial values of the vectors are calculated with the aid of pre-processed matrices in the training phase. For each word in the sequence, the query vector of the computed word is multiplied with each word's key

vector in the sentence (see Figure 13 again). Afterwards, the product is normalised and multiplied with the word's value vector (resulting in z1). The output of the self-attention layer is the sum of all weighted values (hidden state), which is passed on by the Feed-Forward layer. The extracted information can be visualised like in the sentence "I kicked the ball" shown on the right. While "I", "kicked" and "ball" keep important information for the meaning of the word "kicked" in the context, "the" is rather irrelevant.

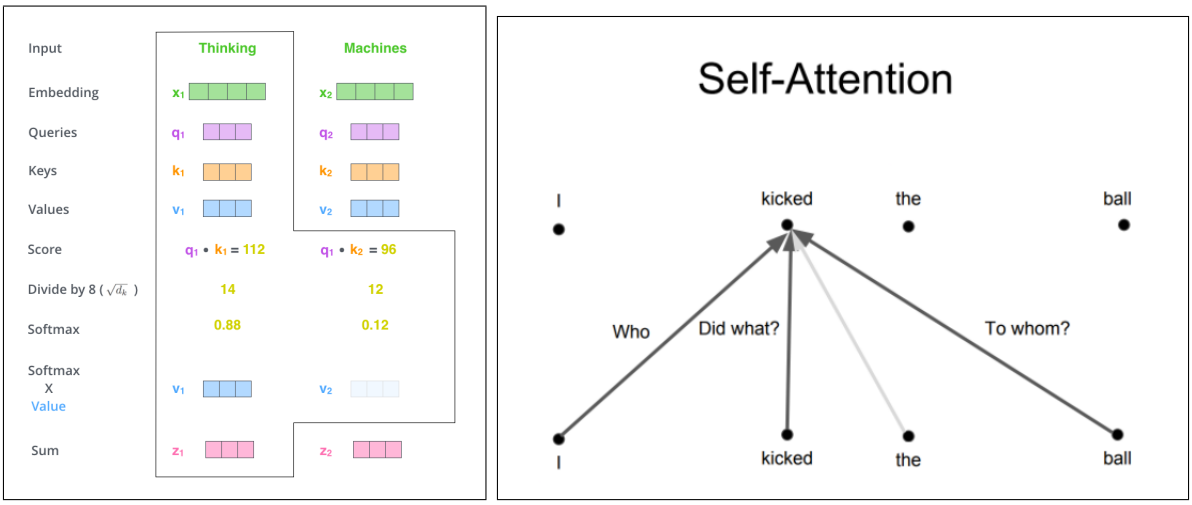

Figure 13: Self-Attention in Transformer Models

Apart from the described components of Transformers, positional encoding of words within sentences is taking place to support a word-order independent computation. Furthermore, within Transformer architectures a multihead attention mechanism is used, which can address multiple queries like "Who" and "Where" at once (Figure 13) or in other words have several query, key and value vectors for each word in the sentences. Transformers make use of eight heads, which are initialized with random values and then trained on the data .[**?**]

Following the order of computation steps, the first decoder receives a vector set from the last encoder, and each word in the sequence is - with the help of its positional vector and the information from the other vectors - decoded through the chain of decoders ending in a final sequence transduction (Figure 11). [**?**]

4.1.3 Transformers for multilingual NLP tasks

For now, Transformer's cross-lingual results are better with a small subset of languages than for a large one. Still, they are a current state-of-the-art for the transfer between languages, and over the last year, research had focused on those architectures by continuously developing new models.[**?**] According to the GLUE benchmark records for English, Transformer

architectures dominate over RNN and CNN solutions for language understanding tasks. Also, in the field of cross-lingual language understanding (XLU), Transformers are mainly used and jointly trained on many languages. [**?**]

There are three dominant, massively pre-trained, multilingual models. First, the masked language model BERT from Google Research has a multilingual version, called mBERT trained on 104 languages [**?**]. Furthermore, Facebook published at the beginning of 2019 the Cross-Lingual Language Model (short XLM) providing different pre-trained models, mainly on two languages but also one trained on XNLI dataset with 15 languages [**?**]. The third model, published later the same year, is a combination of RoBERTa and XLM (XLM-R). It has two publicly available pre-trained versions (12-layer and 24-layer neural network) on 2.5TB on CommonCrawl data of 100 languages [**?**]. All three models are trained without any cross-lingual supervision [**?**](see Figure 7). Next section describes those multilingual models in more detail.

4.2 Multilingual Transformers models

4.2.1 mBERT

Bidirectional Encode Representation from Transformers - BERT, is a 12 layers Transformer neural network, enabling representing words in many languages. Its learning phase for text-classification is subdivided into two main sections. First, the pre-training takes place and is undertaken in an unsupervised manner with a dataset consisting of a huge amount of sentences. Secondly, in the fine-tuning step, labelled data is used to train the model with the parameters from the pre-training step for a specific task (see section 3.3.1). While the architecture of the pre-training phase is similar, it will differ for the specific downstream tasks (see figure 14). [**?**]

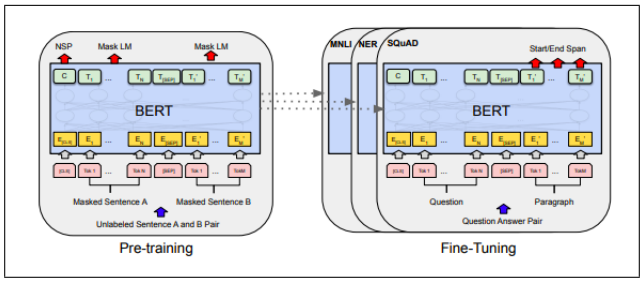

Figure 14: Pre-Training and Fine-Tuning Procedure in BERT

Source: [**?**]

In pre-training phase input words are annotated randomly with [MASK] labels, which

present the words that should be predicted and [CLS] labels at the beginning of each sentence. Usually around 15 percentage of the tokens are marked for prediction [**?**]. Additionally for language understanding task where a relationship between two sentences is relevant a [SEP] label is also used between two sentences to allow next sentence prediction, (see Figure 15). The labeled sentences then pass the encoding Transformer's layers. The hidden state vectors at the end of the pre-training correspond to the masked words. A risk is therefore that the decoding in the fine-tuning phase can just make limited use of the pre-trained language representation when the intersection of masked words and labeled words in the fine-tuning dataset is small.[**?**]

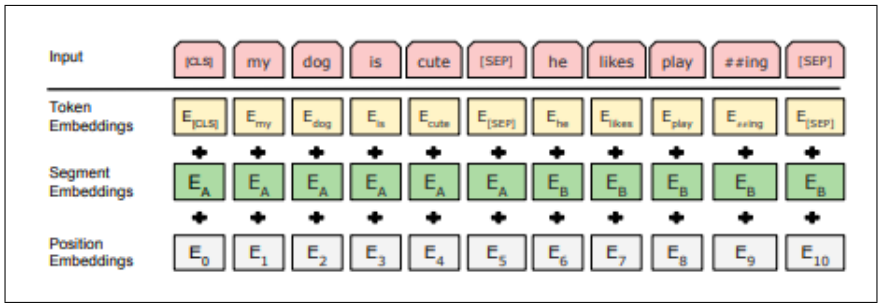

Figure 15: Input Format in BERT

Source: [**?**]

Afterwards, in the fine-tuning, BERT uses the usual self-attention mechanism of the Transformer, which allows fine-tuning for several tasks depending on the annotation in the fine-tuning dataset.[**?**]

BERT's ability to take right and left words around the target word simultaneously into consideration in the learning phase and therefore having both-sided information for studying the semantic of the word, makes it very attractive for multilingual text classification tasks. In contrast, ELMo bidirectional approach uses a feature based extraction, reading first from left-to-right and then right-to-left (with LSTMs) and concatenating the information afterwards . BERT outperforms OpenAI GPT and ELMo on GLUE (in English) on all tasks [6].[**?**]
Multilingual BERT or its abbreviation mBERT is a model jointly pre-trained on 104 languages. The whole training dataset consisted of 120k tokens from raw Wikipedia articles and no aligned word embeddings were used as an additional input [**?**]. The languages can share a common word piece vocabulary across the articles that their training datasets are generated from. Thereby, the input sentences do not contain any language tags. BERT authors claim that it does not require similar representations like a character overlap across the languages. Nonetheless, to overcome the high variation in training data size across the

[6]see evaluation scheme of GLUE here: https://arxiv.org/pdf/1810.04805.pdf

104 languages, the ones with many articles on Wikipedia are sub-sampled, and for languages with a small size of text data, exponential smoothing is used to super-sample them. [**?**] [**?**]

4.2.2 XLM-RoBERTa

XLM-RoBERTa is a blend of RoBERTa and the updated version of original XLM model[**?**]. XLM uses Byte-Pair Encoding as an input for training. The sentences and words are split into the most used subwords in the corpus of all the training languages and the model is trained on them. It is an improvement to mBERT, whose training dataset is not created with a focus on containing the same words across the languages and thus the learning of shared presentations can be limited. Moreover, in the XLM training phase, input samples are given in two languages simultaneously (in mBERT samples are just in one language). It allows the network to predict and learn the missing marked word in one sentence with the information from the other sentence, since different words are marked in the two input samples.
Additionally, XLM's input consists of a language ID and an annotation with information about the order of words in that language. It should support the learning of semantic dependencies. Incorporating the latter two new properties to mBERT's MLM method, the publisher name their model "Translated Language Modeling" (TLM). [**?**]
XLM-RoBERTa enhances XLM by using larger training set - 2.5TB (250k tokens) and by using 100 languages for training instead of 15. The second component - RoBERTa - is a robust version of BERT but trained longer and with larger corpora (by Liu et al.(2019)), achieving better results on next sentences prediction tasks, e.g, question answering.[**?**].[**?**] While mBERT was trained on Wikipedia articles, XLM-RoBERTa was made with the aid of CommonCrawl data from Wenzek et al., (2019)[**?**].[**?**] The number of intersecting languages between mBERT and XLM-R is 88. XLMR base version consists of 12 layers while the large version is built from 24-layers.

4.2.3 Multilingual Transformers for NER

We saw two multilingual Transformers, pre-trained on more than 100 languages: mBERT and XLM-RoBERTa. Both models can be adapted for NER task.
On CoNLL dataset 2002 and 2003 for the language English, Dutch, Spanish and German, the two models reach the following f1-scores:

Model	train	#M	en	nl	es	de	Avg
Lample et al. (2016)	each	N	90.74	81.74	85.75	78.76	84.25
Akbik et al. (2018)	each	N	**93.18**	90.44	-	**88.27**	-
mBERT†	each	N	91.97	90.94	87.38	82.82	88.28
	en	1	91.97	77.57	74.96	69.56	78.52
$\text{XLM-R}_{\text{Base}}$	each	N	92.25	90.39	87.99	84.60	88.81
	en	1	92.25	78.08	76.53	69.60	79.11
	all	1	91.08	89.09	87.28	83.17	87.66
XLM-R	each	N	92.92	**92.53**	**89.72**	85.81	90.24
	en	1	92.92	80.80	78.64	71.40	80.94
	all	1	92.00	91.60	89.52	84.60	89.43

Figure 16: Results on Named Entity Recognition for different multilingual models

Source: [**?**]

The results are also benchmarked against Lample et al. (2016) using a bidirectional LSTM network and a linear CRF output layer (mentioned in the section 2.2.2). Even if semantics of names seem to have more a lexical character, and context is the same (for example when comparing "the Boston Hospital" with "the Hospital of the City of Boston"), the two Transformer models outperform the BiLSTM with CRF.[**?**] The large version of XLM-R shows generally the best results but demands higher memory capacities and longer training times. Comparing mBERT and XLM-R basic version, both having 12-layers, the results go mostly along with each other.

5 Related Work

Achieving complete language-independence for multilingual models as, for now, this problem remains unsolved. The experimental identification of dependencies is the main step for providing approaches how to create language-agnostic systems. XLM-RoBERTa is an example for such a system as experiments showed that a higher vocabulary size resulted in a better generalization across languages (see section 4.2.2).

Nonetheless, generally several publications show different accuracy scores across language combinations [**?**], just a few investigate the serving causes for it. Further, exceptions in the findings show challenges allowing universal assumptions (i.e. Rahimini et al. (2019) shows better results for languages linguistically close but also a high accuracy for the combination of Italian and Indonesian [**?**]).

The chapter is separated into experimental findings across architectures and NLP tasks, and language-independence in multilingual transformer models for NER. At the end we show how our research is positioned within them and how it benefits from previous publications.

5.1 Language-independence in cross-lingual transfer

Languages with available annotated resources allow evaluations of cross-lingual outcomes, presenting for a lot of languages that cross-lingual results are still far away from its monolingual benchmarks.[**?** **?**] Projecting information to another language's sentence is limited by current systems' ability.

Transfer learning within these systems can take place in the general language representations, static and contextual word embeddings (see section 3.1) and for a specific NLP task (see section 3.3). It can therefore be considered on two levels in system architectures: 1) within the generalising across language presentations and 2) within cross-lingual NLP tasks projections, where the first one obviously influences the second aspect.

Studying the choice of a suitable high-resource transfer language for a specific target language and task, Lin et al. (2019) centre their experiments around phylogenetic and typological properties as well as word overlap and data size by using several existing linguistic feature databases as a base (e.g., Glottolog for geographical and genetic distances between languages or WALS for syntactical and phonological distance). The classification model is called "LANGRANK".

	Method	MT	EL	POS	DEP
dataset	word overlap o_w	28.6	30.7	13.4	52.3
	subword overlap o_{sw}	29.2	–	–	–
	size ratio s_{tf}/s_{tk}	3.7	0.3	9.5	24.8
	type-token ratio d_{ttr}	2.5	–	7.4	6.4
ling. distance	genetic d_{gen}	24.2	50.9	14.8	32.0
	syntactic d_{syn}	14.8	46.4	4.1	22.9
	featural d_{fea}	10.1	47.5	5.7	13.9
	phonological d_{pho}	3.0	4.0	9.8	43.4
	inventory d_{inv}	8.5	41.3	2.4	23.5
	geographic d_{geo}	15.1	49.5	15.7	46.4
LANGRANK (all)		51.1	**63.0**	**28.9**	**65.0**
LANGRANK (dataset)		**53.7**	17.0	26.5	**65.0**
LANGRANK (URIEL)		32.6	58.1	16.6	59.6

Figure 17: Contribution of linguistic characteristics for choosing the right transfer language

Source: [**?**]

Their results in figure 17 show, determining linguistic features differ across NLP tasks (here Machine Translation (MT), Entity Linking (EL), Part of Speech tagging (POS) and Dependency Parsing (DEP)). Moreover, the results in MT for choosing the best transfer language are even better when just using dataset-related decision-features.
Phonological features (expressed in d*pho* and d*inv* using different databases) seem to depend on their data source's quality presenting different results based on the database. For EL geographical, genetic and syntactic features play a crucial role. [**?**]
The outcomes demonstrate: identifying universal language-dependencies across tasks is not possible. Further, doing it task-specific depends on linguistic databases' qualities.

For the general language representation, word-order is a present topic across models: The bidirectional LSTM with CRF layer architecture from Xie et al. (2018) (previous state-of-art in NER) working with static aligned word embeddings enriched Lample et al. (2016) model by adding a "self-attention" layer. It introduces a feature vector for the words' positions in the sentences and outperformed Lample et al.'s original model.[**?** **?**]. Ahmad et al. (2019) compares order-sensitive sequential RNN model against non-order-sensitive self-attention model and shows that the latter one outperforms the RNN.[**?**]

The aspects mentioned above concentrate on the idea of understanding the cross-lingual learning of neural networks. Conneau et al. (2020) claim that challenges of generalisation across languages are also within the training process itself, especially for massive multilingual systems. Their learning involves many languages, but the training size cannot be simply increased accordingly - it is limited by the amount of time willing to spend on processing the

data and available space. For this reason, multilingual systems underperform in monolingual results (where system is pre-trained (inter alia other languages) and tested on the same language) in comparison to systems that are just pre-trained monolingual. When upgrading the transformer model XLM trained on 7 languages to 100 languages the f1 score on XLNI dataset drops by 4% [**?**].

Nonetheless, massive multilingual systems are especially attractive and demanded for cross-lingual applications requiring the involvement of several linguistic different languages or low-resource languages - the benefit from the joint training allows their deployment for these purposes, as shown in several investigations [**?** **?** **?**] A trade-off between performance and the amount of trained languages has to be made in the serving of reaching language-independence: Artetxe et al. (2019) shows that in relation to a defined model size just adding a certain amount results in better multilingual generalisation.[**?**]

At the end of the research of this thesis, we came across another very recent publication (from April 2020): Libovický et al. compare the multilingual potential of aligned static word-embeddings (trained supervised with bilingual corpora and contextual word-embeddings (generated by mBERT in an unsupervised manner). Even if static embeddings do not contain semantic properties, the single word embeddings are supposed to be identical across languages. Between matching words in different languages contextual word embeddings present less differences than static embeddings[**?**]. Concluding, they proof to be more language-independent, even if they still carry language specific information. Semantic information seem to reduce cross-lingual biases. Also shown by Wu and Dredze (2019) [**?** **?** **?**].

5.2 Language-independence in Transformer for cross-lingual NER

For the challenges in cross-lingual transfer learning on entity labeling, with the focus on transformer models, mainly *Pires et al. (2019)*, *Wu and Dredze (2019)* and *K et al. (2019)* address it in their experiments, concentrating on BERT's ability of projections across languages in zero-shot settings.

Striving to understand how mBERT makes zero-shot cross-lingual language transfer possible ***Pires et al. (2019)*** use CoNLL-2002 and 2003 and an additional in-house annotated NER dataset. Pires et al. experiments show good results for target languages not sharing the same script with the fine-tuned source language. Also word overlap, being tested on pair combinations of 16 languages, it seem not to be an attribute that has to be considered for mBERT either. In contrast, e.g., in EN-BERT f1-scores depend highly on the percentage of overlaps.
Additionally, the paper indicate the more typological similarities a language pair has in common (measured by WALS features), the better the results for classification tasks, like

NER. Especially, a shared order of words in a sentence seem to benefit the quality of a shared representation between languages in a common vector space: In their experiments for POS-tagging, another text classification task for grammatical properties, the language pairs[7] (EN,Bulgarian (BG)) and (BG,EN) have scores (82.2% and 87.1% respectively) 10 percent points under the monolingual f1 scores. Simultaneously, (EN,Japanese (JA)) and (JA,EN) result in f1 scores almost half of the monolingual f1 scores. In opposite to Bulgarian, English does not share the same order of subject, verb, object with Japanese.
Furthermore, in their experiments mBERT is fine-tuned on POS tagging again with Urdu (Arabic script) and tested on Hindi (Devanagari script), which is not part of the 104 pre-trained languages. F1-scores close to monolingual scores demonstrate mBERT's cross-lingual transfer ability making it also applicable for unseen languages.[**?**] Summing up their work mBERT does create multilingual representation; however, they are flawed with certain language-pairs, for instance, those with different word order.

	SVO	SOV
SVO	**81.55**	66.52
SOV	63.98	**64.22**

(a) Subj./verb/obj. order.

	AN	NA
AN	**73.29**	70.94
NA	75.10	**79.64**

(b) Adjective/noun order.

Figure 18: POS accuracies when transferring SVO/SOV languages or AN/NA languages. Row = fine-tuning, column = evaluation

Source: [**?**]

It can be recognised, that for languages with the structure SOV and noun, adjective (NA) sentence structures are important. [**?**]

With comparable experimental settings for NER (also using CoNLL 2002/2003 dataset and a Chinese dataset), ***Wu and Dredze (2019)*** investigate mBERT fine-tuned on five NLP tasks in English and evaluate annotations outputs on the rest languages. While the results for Dutch, Spanish, German are in a range of 69.56% to 77.57%, for the more distant Chinese f1 score is 51.90%. Firstly, the results are worse in this zero-shot setting than when mBERT is evaluated on the same language it was trained on. Second, the degree of language similarity seems to have a crucial impact, at least for the tested languages, limiting mBERTs general learning of annotations across languages.

A layer-wise consideration of how mBERT's pretraining keeps individual language information tested on 99 languages, shows that at every layer mBERT is able to classify languages

[7](training language, test language)

with an accuracy of 96%.
Their experiments also consider subword overlaps between source and target language in mBERT and in opposite to Pires et al., impacting the performance in NER, POS and Dependency Parsing where the latter two were tested on more languages.
Note that in this experiment mBERT was fine-tuned only on English and the overlap of Dutch, Spanish, German and Chinese calculated with English for NER while Pires et al. experiments tested on a all language pair combinations from a dataset corpus of 16 languages. Nonetheless, there was also a high word overlap correlating with f1-scores for other language understanding tasks tested on more languages but always fine-tuned on English. [**?**]

K et al. (2019) investigate BERTs cross-lingual transfer learning from the pre-training using a bilingual version of BERT by fine-tuning BERT on NER in English and evaluate it on the target languages including Russian, Spanish, and Hindi with the aid of LORELEI news and social media dataset. They consider three categories in their experiment-setups for analysing: linguistic similarities between the languages, architecture of the network and the input/ learning objective.
Interestingly, the results show, in contradiction to Pires et al. (2019) and Wu and Dredze (2019), that no word piece overlap is necessary even in bilingual settings. Their findings were made by introducing a "Fake-English" with shifted characters avoiding an overlap with any other English text (based on Wikipedia as a source).
The impact of structural similarities between language pairs are considered by looking at word-ordering and word frequency. Here, they use again the already introduced artificial English language not sharing any vocabulary/characters and additionally change the order of words randomly and doing the same in the target language. Their findings for word-ordering show that the more the properties of word order differ between Fake-English and the target language, the more f1-scores drop. Word-order seems to be a crucial aspect between the set of pre-trained languages and fine-tuned languages. Moreover, further experiments test bilingual BERT for training unigramm word frequencies show very low f1-scores for NER demonstrating the important information loss for BERT's ability learning cross-lingual representations.
Considering the architecture an learning objective aspect, the number of layers in BERT and its number of parameters can leverage BERT's cross-lingual evaluation and BERT is, as the other publications showed already, able to identify languages without supervision. [**?**] Word-order seems to be again a determining impact on cross-lingual transfer, not just between fine-tuning and evaluating also K et al. presented experiments showing the existing relationship between pre-training and fine-tuning in BERT while word overlap seem to be less influential.

BERT and its multilingual version seem still to absorb language-specific information. A modified version of mBERT, called UDify, was developed to create Universal dependency

treebanks for languages, especially for low-resource ones. The outcomes of this model show again mBERT's strong capability for understand syntactical properties between languages. [**?**])

As mentioned already in the previous chapter, at the end of the research of this thesis, we also came across another recently made publication in April 2020. Next to the comparison between static and contextual word embeddings, **Libovický et al.** look at the multilingual transformer models mBERT and XLM-R as well as UDify and trains DistillmBERT (a BERT model with knowledge distillation [**? ? ? ? ? ? ?**] having 6 layers instead of 12) on the same training dataset as mBERT, checking if less space of representation generalises more strongly representations across languages. Their findings show that feeding the contextual word embeddings the models to a language ID classifier, that the positioning of the languages mainly corresponds to language families. Kudugunta et al. (2019)'s investigations on multilingual presentations across different multilingual neural models confirm the same not just for transformer models by applying as well a language classifier which results show a heavy grouping across 103 languages into their language families.[**?**]. Continuing with Libovický et al., their results shows the strongest language representations, followed by DistilmBERT, UDify and finishing with XLM-R, see figure 19.

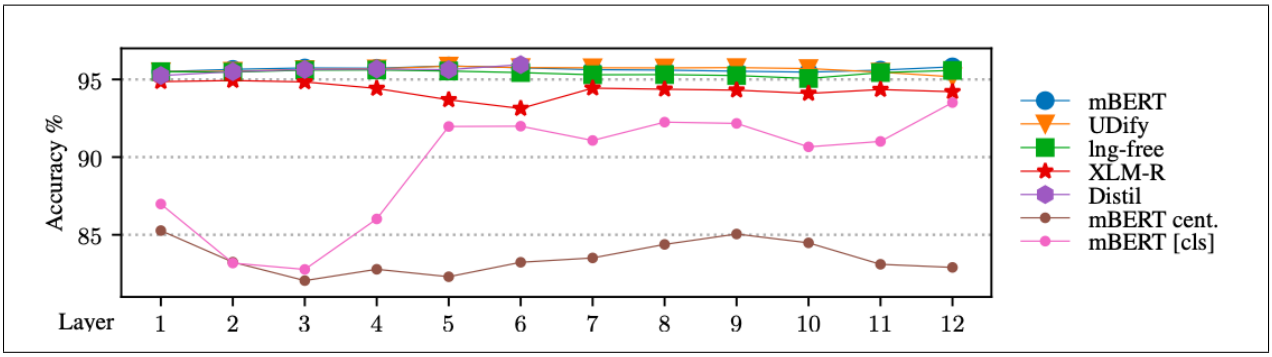

Figure 19: Language ID accuracy for different layers

Source: [**?**]

Libovický et al. expect that the pre-training method of mBERT containing next sentence predicition can be responsible for this which demand stronger language specific knowledge to figure out if two sentences are adjacent. Furthermore, a sentence retrieval score per layer (number of sentences returned based on a specific query) shows that mBERT and XLM-R both perform best at the 8th layer while XLM-R outperforms mBERT at all layers.
We decided to include this publication since it shows a comparison between mBERT's and XLM-R's language-dependence with the results that both mBERT and XLM-R capture

language-specific information in the learning phase while XLM-R outperforms mBERT in this sense slightly.

5.3 Overview and positioning of our work

Pires et al. (2019) claim that mBERT is powerful for zero- shot cross-lingual NER, where languages sharing typological similarities perform best, word order property especially. *Wu and Dredze (2019)* broaden the experiments to more tasks and languages in bilingual BERT settings and they state that projection of language knowledge depends on language similarities, subword overlaps, among others. Further, language-specific information are learnt in all layers. Similarly to Pires et al., *K et al. (2019)* proves - with a modified English language - no influence of word overlap on cross-lingual transfer learning, underlining significant differences in the word order between languages.

Acknowledging that all three works draw attention to BERT/ mBERT model and in most cases use English (or Fake-English) as source transfer language, we go beyond these findings by looking also at XLM-RoBERTa as another multilingual transformer model and investigate the impact of further syntactical properties than word order on Transformers shared by languages using a typological database. We hope showing a more complete understanding of MTM's cross-lingual biases with our experiments to inspire future work for developing massively language-independent transformer models.

Precisely, we:

1. test mBERT also for other language combinations in fine-tuning and evluating in zero-shot settings

2. investigate next to word order also the impact of other syntactical attributes between languages

3. compare mBERT's language-dependence as well against the other multilingual transformer model XLM-R (similar to Libovický et al. we came across later)

6 Experimental setup

We will try to prove or disproof the following hypothesis: mBERT and XLM-RoBERTa are not language-agnostic. As explained in section 3.2.3, the lack of linguistic knowledge encoded in the training dataset, does not ensure language-independence. Thus, we must disprove language-independence experimentally. In this chapter, we first describe ES features, the Wikiann, and the initial experiments we made to justify the, somewhat controversial[8], choice of the dataset. Further, we describe our hypotheses for the experiments and the experiments themselves: the *Language-pairs* experiment and *ES features experiment*. In them, we use mBERT and XLM-RoBERTa, to see if these models incorporate language-specific typological knowledge.

6.1 Typological properties of Indo-European Languages

Linguists classify languages by their typological attributes. The most extensive database available online, providing insights about those typological attributes of language groups, is WALS ("The World Atlas of language structures online") [**?**]. We, however, chose a less extensive Standard Average European (European Sprachbund) feature set of Indo-European Languages. Its size allows us to inspect all features, while still touching on a significant subset of languages.

6.1.1 Twelve features of SAE

The European Sprachbund is a linguistic geographical area characterized by the presence of SAE languages - a region belongs to ES if its spoken language is a SAE language. [**?**]. The features are not represented in most of the global languages; rather they can be observed mainly in the Indo-European languages - a language family of Europe, the Middle East and South Asia, containing Romance, Germanic, Balto-Slavic languages and the Balkan languages.[9] [**?**]. Standard Average European consists of the following features:

1. *definite and indefinite article*, like "the" and "a" in English
2. *relative clauses with relative pronouns*, for example: "The man who drives the car" vs. "The man whose car I drove" in English - "man" is the head of the sentence the pronoun depends on
3. the *have perfect*, a tense using the verb "have" and the *past participle form* of a verb

[8] Wikiann is not man-annotated

[9] But shared features are not applicable to the Armenian, Iranian or Indic branch of the Indo-European.

4. *nominative experiencer*, which indicates that predicates can be general or inverting, e.g., "I like it" may be pointed to an agent, but it could also be "It pleases me" where it is pointed to a goal (inverting)

5. *participial passive*, which is formed by a linking verb and a past participle form of the verb

6. *anticausative prominence*, ADD from research paper

7. *dative external processors*, for example in German "Die Mutter wäscht dem Kind die Haare" direct translation is: "The mother is washing the child the hair." instead of ".. the child's hair". "The child" is the dative and "the hair" is the accusative.

8. *negative pronouns with lack of verbal negation*, e.g, in the English sentence "Nobody came" the verb itself is not negated but rather it is indicated by the pronoun.

9. *particles in comparative constructions*, ADD from research paper

10. *relative-based equative constructions*, for instance in German "so gross wie ich" meaning "as tall as me", the sentence uses a relative adverb "wie" to introduce a relative clause.

11. *subject person affixes as strict agreement markers*, which means that the subject pronouns can not be dropped to give a clear indication of the subject. A counter example would be Spanish, where you could say "(To) Te amo" for "I love you".

12. *intensifier-reflexive differentiation*: in English "self" is not just used as an intensifier, i.e. "The president himself.." also as a reflexsive "He loves himself". But in a lot of languages like Polish ("Sam prezydent.." and "Kocha siebie"), different words are used for the two cases.

All the above twelve features, present syntactic or morphosyntactic characteristics of languages. A minimum of five features is required for a language to be labeled as an average European language. The following map shows the number of shared features between languages:

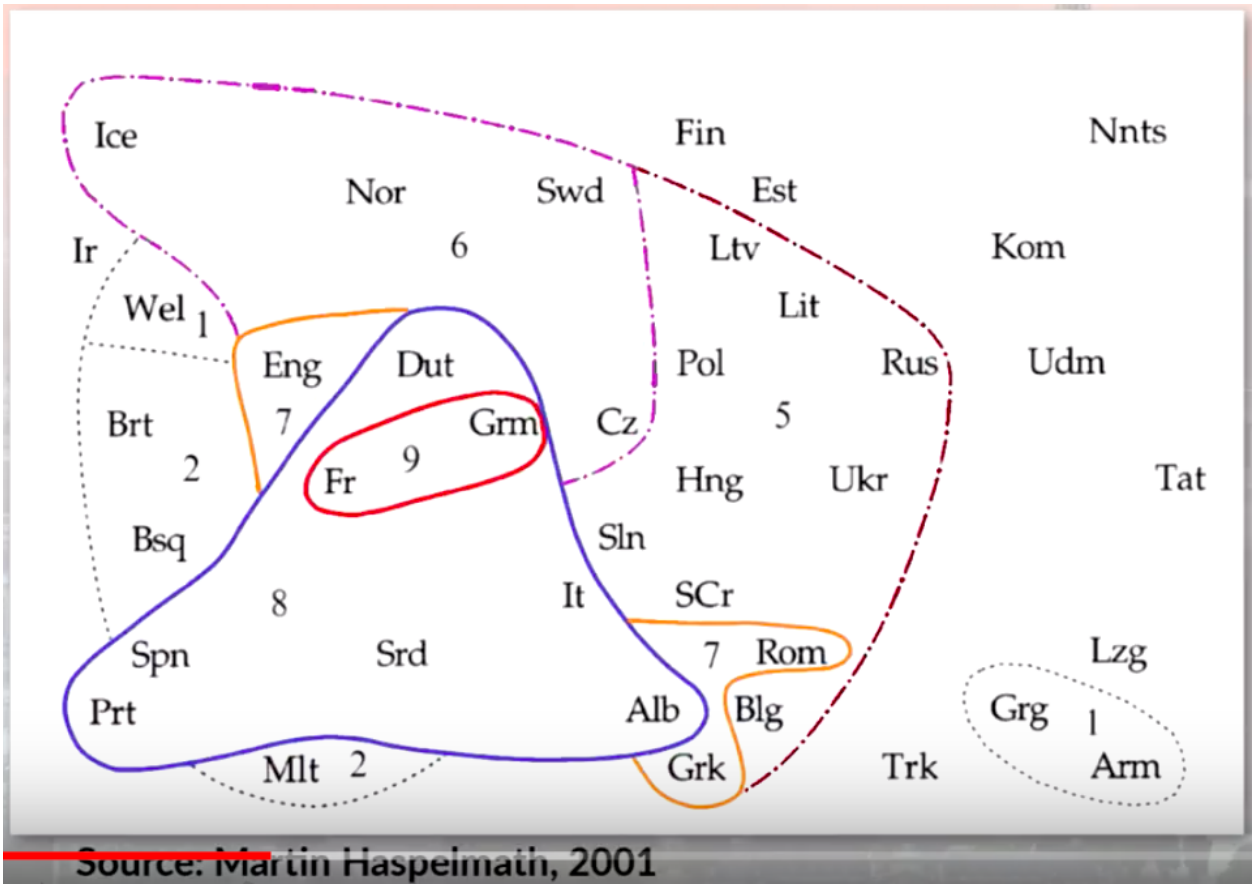

Figure 20: 9 characteristic features of SAE and the number of features exhibited by different languages

[?]

Most of the features are shared by German and French. In general, the Romance and Balkan languages share more features with most of the languages than the nothern ones and English. Slavic languages are most far away from the common SAE features nucleaus.

The well-known rule of a strict order of subject-verb-object across the European languages is not numbered in the European Sprachbund. Additionally, some common features in a lot of other world's languages rarely exist in the European Sprachbund, e.g,. reduplication of words forming for example the plural in Indonesian.

[?]

Basing on (XIV. Typological characterization of language families and linguistic areas,

we created a table in which we may compare Indo-European languages based on their common typological features. We assume that based on running model with one language as a source language (NER trained on it) and other as target language we may derive, based

on accuracy measures, which features are important for our task.

Typological feature	Eng	German	Spanish	Dutch
Definite and indefinite articles	T	T	T	T
Relative clauses with relative pronouns	T	T	T	T
'Have'-perfect	T	T	T	T
Nominative experiencers				
Anticausative prominence				

6.2 Dataset

In natural language processing the best ally we have are not theories and rules but big data (see section 2.1). Languages and differences between them are so complex that it is impractical to describe all the rules manually. For this reason, development of deep learning techniques that endorse a concept of 'The unreasonable effectiveness of data' (the idea that if you have enough data, every model works well) was an enormous push in NLP.

In our research, we use a model that has already learnt languages, so while pushing NER annotated datasets through it, it can focus solely on NER task. Still, it does not mean that the fine-tuning dataset can consist of just a few instances. One of the most popular gold standards for NER is CoNLL-2003 Shared Task. These are manually annotated datasets for English, German, Dutch, and Spanish, proved to have most balanced corpus in terms of named entities classes across other NER sources.[1]

Unfortunately, our experiments require training data from very resource-poor languages, unavailable within CoNLL. On the other hand, manually annotating a few instances, would not be sufficient in size, which is why we studied other sources of obtaining training sets.

6.2.1 Wikiann

Machine translation and language modelling systems' performance grew much faster during the past few years than for other machine learning tasks, like document classification or NER. The reason: training data for the latter is not available *in the wild*.

Fortunately, the vast database of Wikipedia seems to get closest to the perfect source. Cross-lingual name tagging and linking framework[10] using Wikipedia has produced enormous Wikiann datasets. In oppose to CoNLL, they are not human-annotated and thus, they are less reliable in terms of quality. It is, however, indirectly made by humans; it leverages Wikipedia articles and properties of articles which were written and described by people through crowd-sourcing.

Wikiann advantage over CoNLL is in the number of languages (282) and the size of its datasets - up to 200 times larger for English datasets and around 20 times larger for other

datasets (depending on a number of training mentions on Wikipedia). For Indo-European languages, the only unavailable were Lezgian and Nenets. Because our experiment involves Indo-European languages, to use CoNLL (only four languages), we would have to perform annotation projection for each language to provide a training set in a reasonable size. Thus, Wikiann is a more straightforward and cleaner decision. Last but not least, it has the same source as the pretrained model - mBERT was also trained on Wikipedia articles.

6.2.2 Preprocessing

Wikiann datasets are not uniform in length – they depend on a supply of a given language on Wikipedia. To remove biases, we decided to trim datasets to the same size. Firstly, we looked into the sizes in the number of lines and decided to make the cut-off at 100 000 lines. In this fashion, we could still have a significant training size without sacrificing many languages. Thirty-one Indo-European languages remained, and seven were cut-off (Icelandic, Irish, Scots Gaelic, Komi, Maltese, Sardinian, and Udmurt). Breton became the smallest dataset, with around 150 000 lines and 17003 sentences. The remaining languages were then also trimmed to 17003 sentences (chosen randomly). To compare, ConLL's English dataset is 323 555 lines (218 608 training). Additionally, as a form of cross-validation, especially for the most extensive datasets, we created four such 17003 sentences subsets from the original Wikiann. We run all the experiments on all four datasets for each language and take a weighted average of their results. [10]

6.3 Model and Tools

6.3.1 Choice of parameters/Experimental settings

Experimental settings space of deep neural networks is so vast it poses a great challenge for researchers to achieve the best performance possible. Its components include parameters and hyperparameters, which are often, wrongly, used interchangeably. Parameters (e.g., weights and biases) are properties of the training data learnt during training. In contrast, hyperparameters (e.g., learning rate, number of epochs, hidden layers, activation functions, etc.) cannot learn during training but need to be set beforehand. To find good values of hyperparameters, we can perform a hyperparameter exploration, i.e., search across various hyperparameter configurations to find a set that results in the best performance. It is a hard process, given that the search space is vast and brute-force evaluation of each configuration (grid search) is expensive. For that reason, it is also done with a random search or some more advanced techniques, like Bayesian Optimization.

Instead of tuning hyperparameters experimentally, we may also take over the settings from the model we fine-tune on, which often gives good results. If results are not satisfactory,

[10] All the code for preprocessing can be found in take_sentences.py

the learning rate can be tweaked to a slightly lower value. To find if our hyperparameters, taken over from mBERT, are right, we perform tests on ConLL dataset and compare it with benchmarks in the next section.

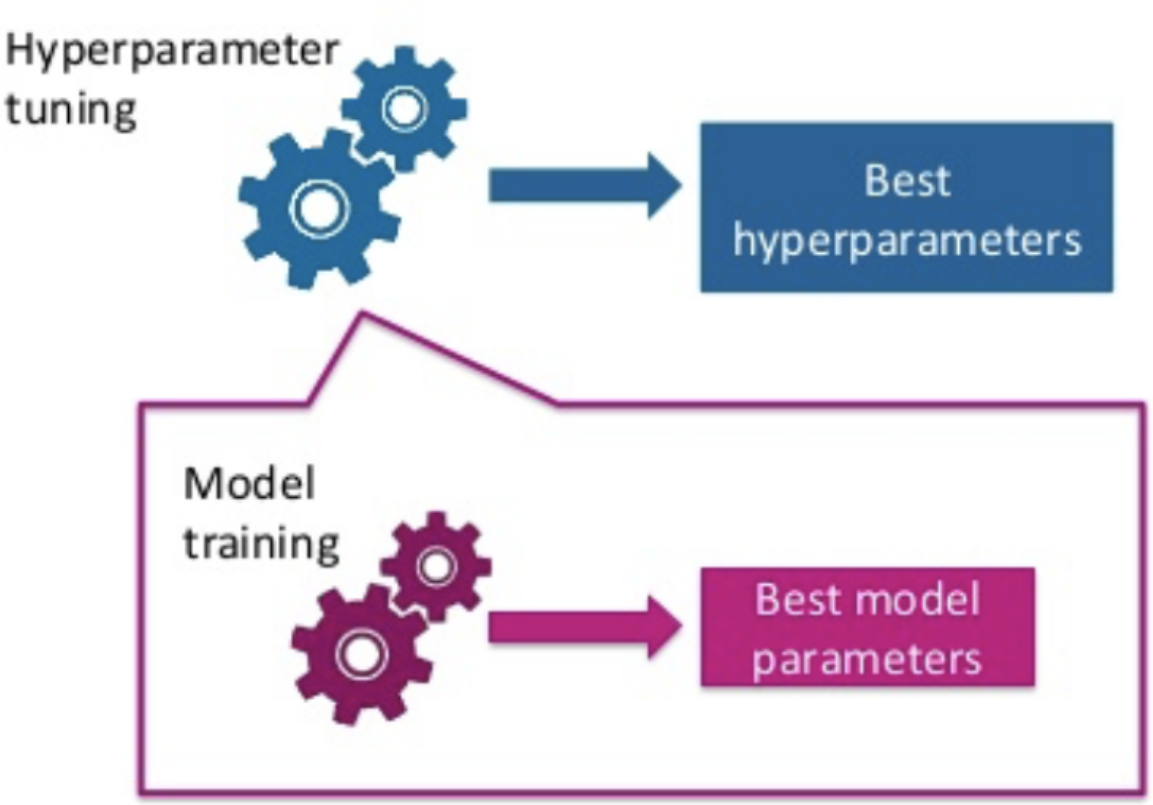

Figure 21: Parameters vs. Hyperparameters

6.3.2 Tools

To fine-tune Transformers for NER task we needed a tool that:

(a) provides Transformers architectures and their pretrained models

(b) enables adding an NER layer on top of mBERT and XLM-RoBERTA

(c) is flexible with the choice of hyperparameters

Two most popular frameworks, both with a big community behind them, are Tensorflow and Pytorch. For our purpose, however, we could look for a higher level APIs, as we only needed the possibility of changing high-level settings, like learning rate or annotation scheme.

Our goal is to compare performance between languages - not finding the best model; so foundation architecture could stay unchanged.

Transformers library[11] from HuggingFace provides architectures together with pr-etrained models, enabling using a given model directly without the pre-training process. For us it means, we don't need to leverage vast corporas of tens of languages to train mBERT - we simply feed NER datasets to the pretrained model. In addition, *Transformers* offers deep interoperability with TensorFlow 2.0 and PyTorch, so potential experiments in further work (e.g., extracting layers) could be added without much trouble. *Simple Transformers* library interface is even more straightforward for our purpose. As a wrapper, build around the *Transformers* library, it abstracts away the complicated setup code while keeping room for the adaptation of the main configurations.

6.3.3 NERModel - code

After creating the environment for *Simple Transformers*, we were able to use its NERModel class. There are two mBERT models available in *Transformers* library: cased and uncased. We chose a recommended cased model trained on 104 languages and consisting of 12-layer, 768-hidden, 12-heads, and 110M parameters. We provided a list of all Named Enitities labels (classes) for the BIO scheme. use_cuda is set to 'true' (by default) to speed up computations by harnessing the power of GPU. To keep mBERT's settings, we leave the attributes in the args dictionary, i.e., model hyperparameters, mostly unchanged (only changing the model and evaluation results saving settings). To the best of our knowledge the only parameter different to original mBERT hyperparameters is the number of epochs. Because Wikiann has lots of training instances and they are similar, we are using one epoch whereas mBERT was trained with three epochs[11].

```
model = NERModel('bert',
                 'bert-base-multilingual-cased',
                 labels=["O",
                 "B-MISC",
                 "I-MISC",
                 "B-PER",
                 "I-PER",
                 "B-ORG",
                 "I-ORG",
                 "B-LOC",
                 "I-LOC"],
                 #use_cuda=False,
                 args={...}
```

[11] https://github.com/huggingface/transformers

6.3.4 Evaluation

We use a hold-out validation technique, i.e., training on one dataset and validating on another. Below, we present exemplary results of fine-tuning the model[12]:

	precision	recall	f1-score	support
ORG	0.78	0.74	0.76	690
LOC	0.94	0.97	0.96	3029
PER	0.93	0.95	0.94	1370
micro avg	0.92.	0.93	0.93	5089
macro avg	0.92	0.93	0.92	5089

Table 1: Exemplary results from eval_results.txt

As we can see in the table, the performance is measured based on the prediction of ORG, LOC, and PER labels. Later, when we compare results, we use the value of micro weighted average to adjust for class imbalance.[13]

6.4 Pre-experiments on mBERT

We run an experiment on CoNLL datasets (Table 8)[14], and compare them against results from the paper: *How Mulitlingual is Mulitlingual BERT*[11] (Table 3), to find out if our hyperparameters are adequately tuned [15].

Fine-tuning/Eval	Eng	Dut	Spn
Eng	**0.95**	0.79	0.73
Dut	0.73	**0.89**	0.72
Spn	0.66	0.67	**0.86**

Table 2: F1 scores for CoNLL dataset

[12]Each model we run was saved in the folder results as pytorch_model.bin file. The evaluation over labels is in eval_results.txt

[13]Micro weighted average aggregates the contributions of all classes to compute the average.

[14]The only preprocesisng to be done was on removing of DOCSTART lines - so it could be combined with Wikiann and the annotation schemes were changed to BIO (to make it equal with Wikiann)

[15]In the appendix: For the purpose of those experiments all Wikiann datasets used in this experiemtns were trimmed to 10% of the original size making it closer to CoNLL sizes **??**

Fine-tuning/Eval	Eng	Dut	Spn
Eng	**0.90**	0.77	0.74
Dut	0.65	**0.90**	0.72
Spn	0.65	0.64	**0.87**

Table 3: F1 score for ConLL dataset from 'How multilingual is Multilingual BERT?'

We obtained comparable results to those from the paper; therefore, we decided to keep the values of the original parameters.

Fine-tuning/Eval	Eng	Dut	Spn
Eng	**0.88**	?	0.64
Dut	0.72	**0.93**	0.86
Spn	0.70	0.84	**0.92**

Table 4: F1 score for Wikiann dataset - large datasets

Next, in Table 3 results from Wikiann datasets are presented. They are close to those obtained on CoNLL's. However, our validation set is not fully reliable as it is simply another subset of Wikiann dataset (hold-out) while the traditional method of evaluation is to compare against human performance. Since there are no manually annotated instances of Wikiann, best we can do is to evaluate it on CoNLL's test set (because it is man-annotated). Those performance results (Table 5) are significantly lower than before.

Much poorer results when evaluating with CoNLL do not necessarily imply poor quality of Wikiann - for most corpora there is a significant drop in performance for out-of-domain training. Because previous results were very good (when evaluated on Wikiann) we suspect the reason of drop in performance are variances between data sources (context variability): CoNLL was trained on Reuters news stories (for English), not on Wikipedia.

Fine-tuning/Eval	Eng	Dut	Spn
Eng	**0.47**	?	0.26
Dut	0.52	0.48	**0.62**
Spn	**0.52**	0.42	0.48

Table 5: F1 score for Wikiann dataset evaluated on CoNLL

6.5 Hypotheses and Design of Experiments

6.5.1 Hypotheses

Based on our previous literature exploration, we present probing experiments testing the validation of the following hypotheses.

Hypothesis 1: Fine-Tuning and Evaluating mBERT on the same language show better results than doing the same on different languages This hypothesis tests mBERT's general ability for its language-independence. From previous research and also our pre-experiments on the CoNLL-dataset, we could see that cross-lingual f1 scores are still below monolingual f1 scores. Since this is also the main motivation of our research, we expect to observe this phenomena also with our chosen dataset.

Hypothesis 2: Language pairs, which share several ES-features overperform language pairs with fewer shared features Hypothesis number 2 corresponds to the previous papers investigating the multilingual functionality of mBERT for several tasks with the outcome that mBERT seems to be sensitive to typological similarities, inter alia syntactical properties, between languages while lexical overlap plays a secondary role. We then expect to see the same results for our Wikiann datasets, based on the typological features describing the main linguistic characteristics of Indo-European languages.

Hypothesis 3: Evaluating on languages with a feature, gives better results for models fine-tuned on a group of sentences containing a given feature than a group of sentences(languages0 not containing that feature at all If we prove this hypothesis then it is possible that we can have an insight about where mBERT still learns language specific feature for the entity labeling. Since the 12 typological features from the European Sprachbund does not contain word order characteristics of the language, we make in these experiments also use of the database WALS, which present typological features for a huge number of languages, to see which word order properties the used languages have. We want to verify our experiment design and compare it with the findings from chapter 3, showing the impact of word order in mBERT for cross-lingual NER tagging with CoNLL dataset.

Hypothesis 4: We expect the same results for the previous experiment 1 and 2 using the other introduced transformer model XLM-RoBERTa The base model of XLM-RoBERTa is, like mBERT, trained on a 12-layers transformer models with 8 heads, 728 hidden states and sequence length of 128. From previous research, we could see that XLM-RoBERTa performs similar for NER tagging on CoNLL dataset as mBERT. We expect therefore also similar results for the Wikiann dataset, even if there seems to be a higher overlap of words in pre- training and fine-tuning phase for mBERT since both are done with Wikipedia, XLM-RoBERTa is trained on a larger dataset and incorporates the Translated Language Modeling technique by using Byte-Encoding and languages ID (see chapter Transformer Models). If this is true, we also aim for redoing the same feature experiments done with mBERT with XLM-RoBERTa to see if both models show differences in the f1-score the same features or if the different pre-training and architectures can have an influence on the

cross-lingual transfer.

6.5.2 Design of Experiments

All of our experiments were undertaken under zero-shot settings, where mBERT has not been fine-tuned on the evaluated language - just for comparison in monolingual settings. Furthermore the determined values for the parameters, described in chapter "Parameters" remained the same over the experiment design points. Also when tested in Hypothesis 4 for XLM-RoBERTa, we use the 12-layers base version and fine-tune it with the same parameters as for mBERT.

How the design points were chosen for the each experiment setup will be explained in the next sections.

First, language experiments (hypothesis 1 and hypothesis 2) are meant to discover if mBERT is language independent. Next, feature experiment (hypothesis 3) tries to find out which features have impact on the Transformer, i.e., making it language-dependent.

6.6 Experiment typological similarity between languages

6.6.1 Experiment domain

We are going to test the effect of typological similarity derived from European Sprachbund between languages on mBERTs fine-tuning for NER (Hypothesis 1/2). For that we are going to fine-tune and evaluate for language pairs sharing a lot of typological characteristics with each other and language pairs sharing just a few or no features.

6.6.2 Choosing experiment groups

Language pairs were identified by their amount of shared features, using euclidean distance measurements between the language vectors. Those vectors are binary vectors based on the European Sprachbund features - each language has 12 dimensions, with 1s if they have a given feature and 0s if they do not. For each language, language pairs were chosen based on there similarity measurement.

6.6.3 Design points

We run the experiments on 0, 1, 4, 7, 10 and 12 shared features. For each number 20 language pairs (if existed, otherwise the available amount of language pairs) were randomly selected.

6.7 Experiment Impact of individual typological features

6.7.1 Experiment domain

After considering the general dependence of typological characteristics of languages on mBERT's cross-lingual learning, we would like to have a closer look at the impact of individual typological features from the European Sprachbund.

6.7.2 Choosing experiment groups for 12 European Sprachbund features

To improve the quality of the experiments we are not simply taking all languages, but instead choose languages that are most similar to each other, i.e., share most values for the features (see previous section), apart from the one we analyse in each data point. To achieve that, we remove the investigated feature from the language vectors, and represent them by the remaining 11 features in a common vector space[16]. The closest k languages are identified by calculating for each language the k closest neighbors with the aid of a KDTree based on euclidean distance. We decided for setting k to 10 by default, which gives us across all features an average euclidean distance between the 10 chosen languages of around 0.8, which means that on average the chosen 10 languages show except in the considered feature a similarity in 0.6 of the other features. See example for feature 1 in Table ??. There were, however, a few exceptions, where setting k to 10 did not result in two separate languages groups - one having the considered feature and the other one not. In those cases, k was set to 15 and the average euclidean distance was around 1.1 (indicating a rest dependency of 1.2 features except the investigated one). [17]

6.7.3 Design points/Experiment description

In our experiment design we do fine tuning twice - first on a mix of languages not having the feature - *false* group (all features in Europen Sprachbund are binary) and evaluating on a mix of languages having that feature - *true* group. In the second fine-tuning, we add *false* languages to the training set, while evaluation set stays the same. As a result, we have one model trained on languages having the feature and one not having the feature. As stated in hypothesis three, we expect the f1 score to rise for the second fine-tuning if mBERT is language-dependent. Otherwise - it's agnostic, at least in that feature space. Moreover, for each fine-tuning we have two experiments - swapping *true* languages from evaluation set with *true* languages from training set. Each is further done for 4 different lanaguage datasets. Effectively, we have 208 design points - 4 datasets * 2 experiments * 2 fine-tunings * 13 features. Code for gathering all those results into a csv in repo:

[16] See how the vectors were created in the previous section

[17] To see the complete language groups look appendix (?) or our repository finetuning_BERT/DOE:script_generator/doe_features_overview_wiki

Features – results/features_gathering_results.py

In order to make our experiments clean, we impose the following rules:

(a) Each fine-tuning training set has the same size 17003 sentences

(b) Each evaluation set has the same size - 10% on training set so 8502 sentences

(c) To create a language group, language datasets of 17003 are first concatenated and then sentences are read and shuffled; as a result languages should have similar contribution. All shuffling in the code is given a seed so experiments are reproducible.

(d) In the second fine-tuning TRUE and *false* languages are in 50 to 50 proportion for the equal contribution.

In the scripts for experiments there is a one file for each feature. There, we firstly take k languages and concatenate them into following language groups: - all *false* - evaluation (from *true* group) - True no eval (from *true* group excluding languages taken for evaluation)

Based on those 3 files of concatenated sentences (in the random older) we can run all experiments.[18]

true lan.	*false* lan.	Eval. 1	F-T 1.1	F-T 1.2	Eval. 2	FT 2.1	FT 2.2
es	pl	es	pl	pl	ro	pl	pl
pt	sl	pt	sl	sl	sq	sl	sl
it	uk	it	uk	uk	hu	uk	uk
ro	lv		lv	lv		lv	lv
sq	lt		lt	lt		lt	lt
hu	sr		sr	sr		sr	sr
	bg		bg	bg		bg	bg
				ro			es
				sq			pt
				hu			it

Table 6: Language groups for feature 1 - experiment 1 and 2, k = 15

.

Above we have an example of a table from the experiment design for feature number one. In first two columns we have languages having the feature and not having the feature,

[18]Scripts are in the experiments folder, for each feature in a separate directory (e.g., for feature 1 the path is: experiments/feature1/run_transformers.py).

respectively.[19] Next three columns are part of the first experiment: the evaluation set, first fine-tuning training set and second fine-tuning training set. First fine-tuning are always simply all *false* languages. *true* languages are divided between evaluation set and second fine-tuning. In the second experiment we only swap *true* languages - the one that went earlier to second fine-tuning are now the evaluation set and vice-versa.

[19] We are not using all available languages - look in previous section.

7 Results and Analysis

7.1 Language-pairs experiment

7.1.1 Monolingual results and correlation on Wikiann dataset

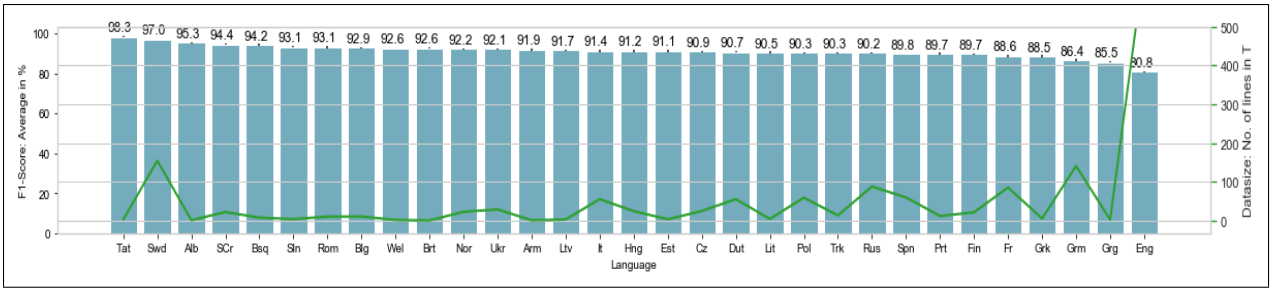

Figure 22: Monolingual Results for mBERT and Wikiann dataset sizes (green line)

First, in figure 22, we present monolingual f1 score for each language. All the results in this chapter are averages of four different random subsets of 17k sentences from the original Wikiann [20]. As indicated by the black line at the top of the bars, all languages have a small standard deviation over those four iterations, which supports the validity of our results. In general, the monolingual f1-scores do not differ a lot along the languages.

Interestingly, English performs slightly worse in comparison to other monolingual results while Tatar's result is the best. Since Wikiann is retrieved from Wikipedia, variations in original datasets' sizes (green line) might be an explanation. The amount of Wikipedia articles in English is much bigger than Tatar, which entails a higher variety of authors and topics[21]. In case, we would have focused on training in English, parameters like number of epochs or batch size, could have been adjusted in mBERT, to reach better monolingual results in English but also need more training time [22]. In the following feature experiments English is used only in one language group, to reduce the effect of the data size inequality on the results.

7.1.2 Cross-Lingual results

Now, we are going to look at cross-lingual results. We repeat the experiment from our pre-experiments on the CoNLL languages (English, Dutch, Spanish) but this time instead

[20] see section 6.2.2

[21] See multilingual Wikipedia statistics: https://en.wikipedia.org/wiki/List_of_Wikipedias

[22] i.e. Lauscher et al. (2020) reaches with 10 epochs and batch size 128 92.3% for monolingual English with Wikiann NER

of running it on a large dataset, we continue with our chosen experiment design of four iterations with dataset subsets of 17k sentences.

Fine-tuning/Eval	Eng	Dut	Spn
Eng	**80.8**	81.4	66.2
Dut	71.3	**90.7**	85.0
Spn	70.0	83.1	**89.8**

Table 7: F1 score for Wikiann dataset for CoNLL languages

Except for the language pair Eng - Dut, all cross-lingual results are below the monolingual counterparts. The highest difference appears when evaluating a model trained in English on Spanish sentences. All in all cross-lingual results are quite competitive; however, we need to take in consideration that the three languages are very closely related - we can see lower f1-scores for more distant languages in the successive experiments. Additionally, we see from this experiment that trimming the dataset to 17k was a good design decision - results are very similar to those from 6.4 section, where we took much larger subsets of Wikiann[23].

7.1.3 Correlation between shared ES features and the quality of the transfer

Here, we address the challenges of cross-lingual projection by using typological features from European Sprachbund. The relation is shown in figure 23 and results in figure 24.

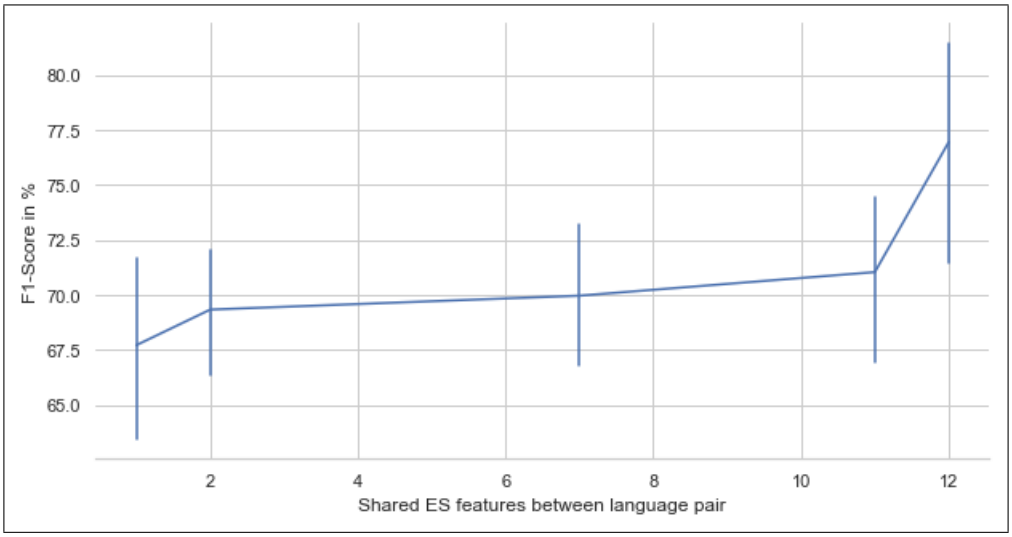

Figure 23: Correlation between the number of shared ES features and the quality of the transfer between language pairs.

[23]see table4

Shared ES-Features	F1-Score Mean	F1-Score Standard Deviation	std_over_4_it	number_of_pairs
12	76.94	10.74	1.34	17
11	71.03	11.80	1.56	36
7	69.96	10.29	1.65	39
2	69.33	8.06	1.44	30
1	67.72	8.34	2.12	14

Figure 24: Result table language distances mBERT

We plot on the x-axis the number of shared ES-features between the languages in the pair and on the y-axis the average f1-score from all the language pairs computed for the amount of shared features. **The graph shows the more features the languages share, the better the overall f1-score**. The vertical bars present the standard deviation of the language pair's f1-scores. From the table, we can see an average of around 9.8%. The standard deviation is quite similar for every feature number, supporting the reliable experiment representation of an existing correlation.

7.1.4 Case Study of a non-pretrained language Tatar

From the thirty-one languages we included in our experiments, Tatar is the only language mBERT was not pre-trained on. Findings within the literature review showed that mBERT performs well for languages it is not pre-trained on (in zero-shot settings). We want to see if the same applies for Tatar in our experiments by looking at the results of language pairs containing Tatar (Figure 25).

shared ES-features	LAN_1	LAN_2	std (over 4 iter)	dist f1 std (over 4 iter)	Δ mean
11	Wel	Tat	5.41	1.27	-7.43
11	Est	Tat	2.07	1.27	5.70
11	Tat	Trk	2.37	1.27	-3.51
11	Arm	Tat	3.44	1.27	-0.19
11	Tat	Arm	2.72	1.27	-15.10
11	Tat	Est	0.62	1.27	3.87
11	Tat	Wel	0.70	1.27	-20.40
7	Tat	Ukr	1.17	1.65	-11.60
2	Eng	Tat	6.20	1.44	-6.70
2	Tat	Dut	0.58	1.44	1.42
2	Dut	Tat	4.51	1.44	1.32
2	Tat	Eng	2.13	1.44	-18.10
2	Tat	Grk	0.84	1.44	-3.28
1	Grm	Tat	4.44	2.12	2.63
1	Tat	Grm	2.09	2.12	-0.88
1	Tat	Fr	1.70	2.12	-3.73
1	Fr	Tat	5.92	2.12	-3.02

Figure 25: Results Tatar

LAN_1 represents the fine-tuned language and LAN_2 the evaluated language. Δ mean is calculated by the difference of the language pair's f1-score with the average f1-score of the distance the language pair belong to, i.e. the distance of Tat, Trk is 1.0 and the average f1-score of this distance over all language pairs is 71.03.

For this case study, we consider Δ mean: the f1-score of each language pair with Tatar and the corresponding f1-score for the language pair's distance (shared ES features between the two languages). Further, from the subsection before, we could see a shared standard deviation over the f1-scores between the distances of around 9.8%. This is our benchmark when evaluating the difference between the two f1-scores in both directions (positive/negative). The highest standard deviation over all experiment language pair show Eng - Tat, followed by Fr - Tat. Overall, projecting from from another language to Tatar results in a higher standard deviation over the four iterations than the other way around (see i.e. Tat - Fr). A possible reason can be the big variation in fine-tuning dataset sizes, and corresponding disproportion of information: Detecting entities in the Tatar dataset during evaluation depends stronger on

the learnt entities during fine-tuning than the other way around. There may be a dependency on the overlap of topics between the fine-tuning and evaluating subset. The other way around - projecting entities in Tatar to another language - this dependency does not seem to have the same impact. Either the quality of the Tatar's dataset is covering a high diversity of topics or what seems more reasonable considering the representation of Tatar on Wikipedia, the pre-training language understanding leverages up consistently the little entity knowledge provided by Tatar for another language.

Moreover, considering now Δ means, Tat - Wel and Tat - Arm show results above the benchmark and therefore another f1-score than other language pairs sharing the same distance. In both cases the languages are represented by small datasets, the topic of the underlying Wikipedia articles [24] can differ a lot as well as writing style since Wikipedia is a public encyclopedia [**?**].

Except the two cases, the f1-scores go mainly along with other language pair f1-scores from our experiments. Therefore our experiments with Tatar prove principally the findings from the literature review, that mBERT also works well for languages mBERT is not pre-trained on. But adding from our results, that results are more consistent when the not pre-trained language is used for evaluating rather than for fine-tuning and evaluating it for a pre-trained language (at least for Tatar).

7.1.5 Case Study: Performance across different scripts

Another aspect, we look at is the impact of different scripts, i.e., when two languages do not have a character overlap. This case study is derived from the idea of the Masked Language Model mBERT is using, making mBERT independent from character representations. The languages from our experiments involve the following alphabets: Latin, Greek, Arabic, Armenian, Cyrillic. We filtered for the language pairs that do not share the same script and presented, for each scripts, all possible combination with other scripts, and their corresponding f1-scores in Figure 26.

[24] see multilingual Wikipedia statistics: https://en.wikipedia.org/wiki/List_of_Wikipedias; small Wikipedia datsets were boosted within Wikiann generation, see section Dataset

script_LAN_1	script_LAN_2	std (over 4 iter)	dist f1 std (over 4 iter)	Δ mean
Arabic	Armenian	2.720000	1.270000	-15.100000
	Cyrillic	1.170000	1.650000	-11.600000
	Greek	0.840000	1.440000	-3.280000
	Latin	1.285556	1.628889	-0.777778
Armenian	Arabic	3.440000	1.270000	-0.190000
	Cyrillic	2.310000	1.650000	4.320000
	Georgian	0.810000	1.270000	-7.330000
	Latin	2.722857	1.642857	-8.480000
Cyrillic	Latin	1.237500	1.287500	2.371250
Georgian	Armenian	1.430000	1.270000	-22.900000
	Cyrillic	0.420000	1.650000	-4.930000
	Greek	0.570000	1.440000	3.040000
	Latin	2.644000	1.532000	-1.290000
Greek	Georgian	0.720000	1.440000	-0.050000
	Latin	4.700000	2.120000	-9.410000
Latin	Arabic	3.521111	1.628889	0.923333
	Armenian	2.300000	1.682000	-21.878000
	Cyrillic	0.945000	1.382500	-0.485000
	Georgian	0.873333	1.453333	-2.853333
	Greek	0.655000	1.885000	7.405000

Figure 26: Results different scripts

Grouped by script language pairs. The f1-scores represent the mean over the language pairs within the group. The difference of the mean for each language pair is calculated by the average f1-score of the distance the language pair belong to subtracted by the f1 score of the language pair.

Our evaluation of the results is proceed in the same way as for Tatar: Δ mean represents the difference of the f1-score of the script language pair and the f1-score mean of the corresponding distance of the language pair and our benchmark is again the average standard deviation of around 9.8% across all distances.

The script combinations Arabic - Armenian, Georgian - Armenian, Latin - Armenian are with differences around -15%, -22% and -21% outside the usual standard deviation. No

script combination showed unusual results in both directions. The left combinations are mostly within the standard deviation of 9.8% or slightly above. Generally, our experiment results seem to be mildly affected by different scripts across the chosen languages.

7.2 ES Features

Based on the correlation between f1-scores with the amount of shared features from European Sprachbund for NER, we further look at the features in isolation, based on the experimental setup described in the previous chapter (section 6.7.3). Fine-Tuning 1 presents the fine-tuning of mBERT on languages, that do not have a given features (FALSE) and evaluate on languages that have the feature (TRUE). Fine-Tuning 2 has the same setup but one: mBERT is fine-tuned on a combination of FALSE and TRUE languages (referred to as MIXED). Further, for each feature we have two experiments using different combination of TRUE group (Experiment 1/Experiment 2).

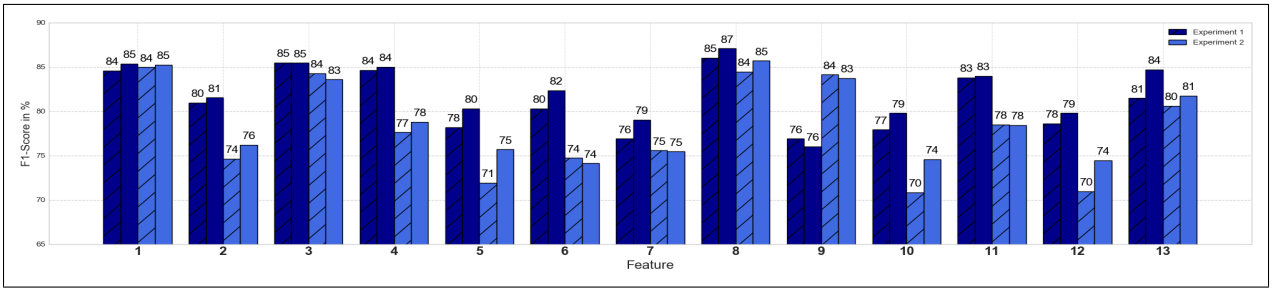

Figure 27: Features (F1-Scores Fine-Tuning1 and Fine-Tuning2)

Marked bars correspond to Fine-Tuning 1 (FALSE to TRUE), and the other one to Fine-Tuning 2 (MIXED to TRUE)

Figure 27 shows the resulting f1-scores for both experiments. Overall, we notice in both fine-tunings higher f1-scores in comparison to the results with language pairs. **Multilingual BERT seems to benefit from being fine-tuned on more than one language in zero-shot settings.** - the advantage of joint training effect already mentioned in the context of pre-training multilingual language representations also seems to affect the fine-tuning step (see section 5).

Proceeding by looking closer at the histogram, we can see that not in all cases experiment 1 and experiment 2 show similar results for the f1 scores. For example, for feature 2 or feature 6, experiment 2 shows significantly lower f1-scores than experiment 1. A possible explanation can be that other linguistic dependencies between the languages, except the considered feature, impact the performance. Nonetheless, we can recognise that fine-tuning 1 and fine-tuning 2 are quite consistent within each feature's experiments. We focus therefore more on the learning effect presented in Figure 28.

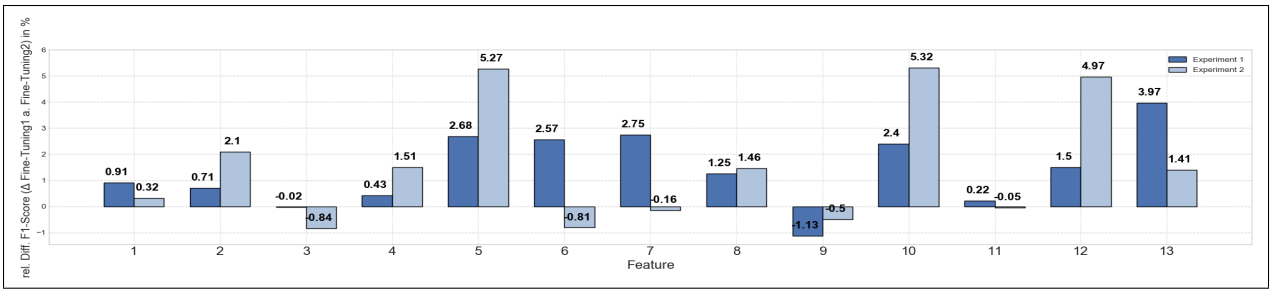

Figure 28: Features (rel. Difference F1-Scores Fine-Tuning1 and Fine-Tuning2)

Comparing the difference between fine-tuning 1 and fine-tuning 2, some features do not show a common increase or decrease in both experiment settings (for example feature 3 or feature 6). In these cases, we can conclude that mBERT is not learning the linguistic property represented by the feature. Feature 9 shows in both experiment a decrease. It seems that adding language, that share the feature to the fine-tuning language group worsen the results, but it has to be noted that the decrease is small.
Across the results, we can see that feature 5, 8, 10 and 12 show the highest increase from the ES features (feature 1, 2 and 4 show in experiment1 or 2 an even higher increase than feature 8, but is less stable over the two experiments). They present the following syntactical linguistic attributes:

Feature N	Typological Feature
5	Participal passive
8	Negative pronouns and lack of verbal negations
10	Relative-based equative constructions
12	Intensifier-reflexsive differentiation

Running our experiments also on word order (positioning of adjective, noun within a sentence) confirms the findings from *Pires et al. (2019)*, *Wu and Dredze (2019)*, *K et al. (2019)* showing that mBERT learns word order representations.[25]
Furthermore, across the results for the features, we can rarely observe huge increases or decreases. The range of the learning goes from -1.31% to 5.32% and showing rather increases than decreases, demonstrating mBERT might still be learning syntactical properties but generally, considered those little improvements, seem to be close to being language-agnostic (at least for the ES features).

[25]It is feature 13 in our repository

7.2.1 Repeating language pair experiment with 4 identified features

In the previous section, our results show a correlation of mBERT's accuracy for cross-lingual NER tagging with feature 5, 8, 10 and 12. We repeat our experiments for the language pairs, reducing the representation of the languages on these four features. Each experiment design point consist again of 20 language pairs and the f1 score mean was calculated for them. The results are shown in Figure 29.

	shared features	f1_mean_over_lan_pairs	f1_std_over_lan_pairs
0	4	69.983616	2.043196
1	2	75.037515	1.404578
2	0	70.484904	1.674684

Figure 29: Language pair results with filtered features

Surprisingly, we can see that language pairs sharing all four features do not outperform languages sharing two or non of the features. There have to be other linguistic attributes influencing the cross-lingual results from the chosen language pairs. We could see from Lin et al. (2019), that next to syntactical linguistic attributes, also geographical or genetic affiliation seem to play a role for NER.

7.3 Differences across multilingual transformer models

Now we are going to consider our results from mBERT in comparison to XLM-R - the other pre-trained multilingual transformer model - conducting the exact same experiments.

7.3.1 Monolingual results

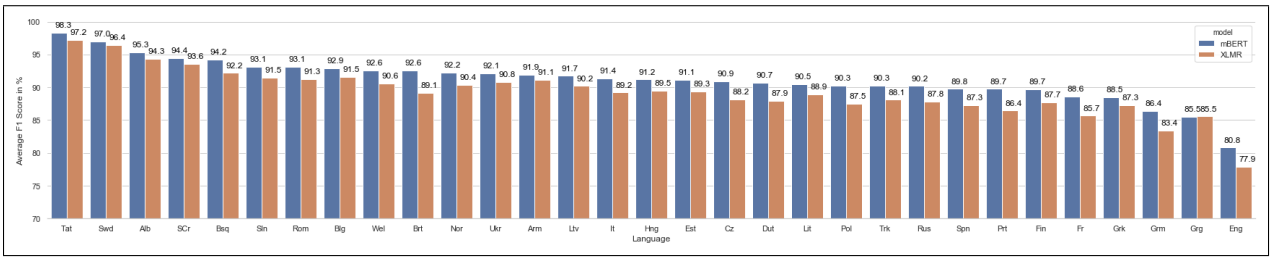

Figure 30: Monolingual Results mBERT and XLMR

For the monolingual experiments, we can also see for XLM-R a correlation of fine-tuning dataset size and f1-scores. English is still the language with the worst f1-score in comparison to the other languages. In almost all cases XLM-R scores are lower than scores for mBERT. A possible reason is that mBERT is trained on articles from Wikipedia (so the same source as Wikiann dataset), while XLM-R is pre-trained on the CommonCrawl. On the other hand, XLM-R contains a much bigger amount of tokens trained on.

7.3.2 Language-pair experiment

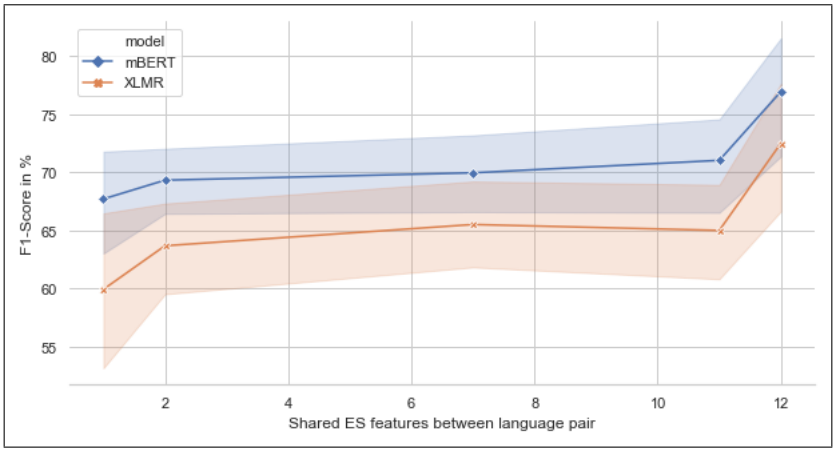

Figure 31: Language Distances Results mBERT and XLMR

For XLM-R too, there exist a correlation between number of shared ES features and the f1 scores of language pairs. XLM-R seem to have more difficulties to transfer between languages that share none or just a few linguistic features in comparison to mBERT. But also for language pairs having several feature values in common, XLM-R results fall behind those of mBERT, what was already visualised in the monolingual results. Moreover, XLM-R has a slightly higher standard deviation over the f1 scores with an average of around 11.7% (mBERT 9.8%).

7.3.3 ES Features

Since the experiment for the language pairs suggests that XLMR learns at least a subset of the ES features, influencing the cross-lingual transfer for NER between the languages, we are redoing the same feature experiments as for mBERT and considering, if XLM-R shows similar impact of certain features.

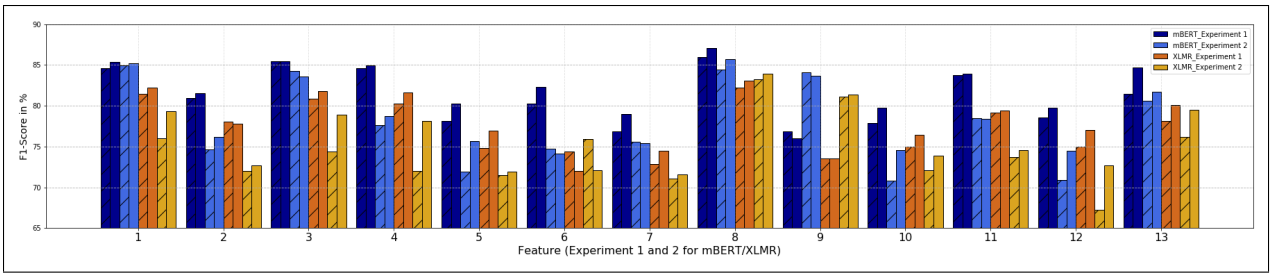

Figure 32: Features (F1-Scores Fine-Tuning1 and Fine-Tuning2) mBERT and XLMR

mBERT in blue and XLMR in orange; from right to left Fine-Tuning 1 (dark color) to Fine-Tuning 2 (light color); each two times - Experiment 1 and Experiment 2.

First, XLMR f1 scores consistently fall behinf the f1 scores of mBERT, which was already observed for the language pairs. Moreover, similarly to mBERT, XLMR's overall accuracy is growing when XLM-R is fine-tuned on more than one language (in Figure 32 results are mainly below 70% mark).

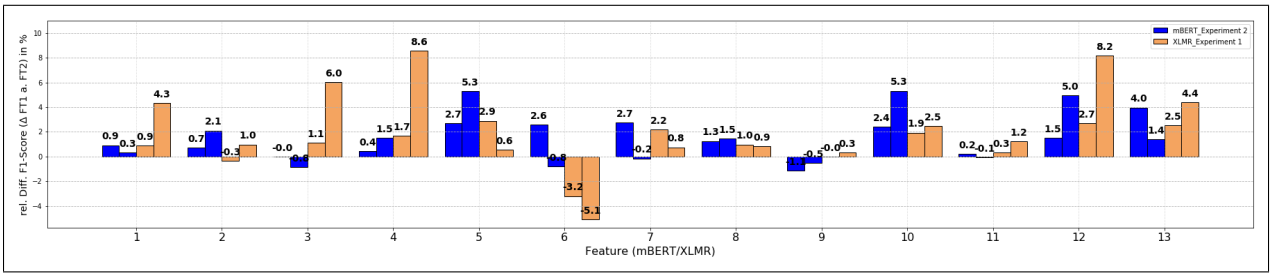

Figure 33: Features (rel. Difference F1-Scores Fine-Tuning1 and Fine-Tuning2) mBERT and XLMR

XLM-R and mBERT accuracy results are matching for feature 5, 8, 10, 12 and 13. Additionally, feature 7 shows clearer in XLM-R that the feature is learnt by the Transformer model. We can also see that XLM-R shows significant increases for feature 1, 3 and 4, but unequally for both experiment - stronger for experiment 2. The results difference between mBERT and XLM-R is most stark for feature 6. Here adding a language, that have the feature to the fine-tuning language group drop the f1-results in both experiments for XLM-R while mBERT shows differences between experiment 1 and 2.
Even if our last experiments from mBERT features reducing the language representation just on the four features show that there must exist other linguistic dependencies influencing the cross-lingual results, we find it interesting, that XLM-R seems to learn the same features as mBERT, underlining an impact of these linguistic properties on cross-lingual NER tagging with transformer models. XLM-R shows a slightly less stability in the results. A possible

reason could be the lack of overlap between pre-training source (CommonCrawl) and fine-tuning source (Wikipedia). Experiments from Wu et al. (2020) show: unmatching domains in the pre-training have a higher effect on the performance than in fine-tuning.[**?**]

8 Conclusion

8.1 Summary

This paper investigated the ability of cross-lingual transfer between languages by multilingual transformer models. This area of research is important because cross-lingual systems become very salable since the number of languages online grows. The sources of this capacity are twofold. First, a single architecture is reused for many languages in oppose to having custom technologies for every single language. Second, if languages are represented in a joint language space, we do not need to develop costly datasets for resource-poor languages, i.e., we overcome dataset scarcity. The early Direct Transfer models trained on many languages learnt little in comparison to their monolingual versions. While massively multilingual trained Transformer results are much more promising in finding language-independent representations across languages, they still do not achieve the same results as they are trained monolingual. We suspect a reason for it is that they incorporate language-specific information challenging cross-lingual projections.
By running two experiments on 31 language NER datasets with pretrained multilingual Transformer models, we first proved with a language-pairs experiment that Transformers are not language-agnostic on Indo-European languages and further there exist a correlation between syntactical linguistic features and their performance. Moreover, in features experiment, we identified four features that, if present in the training set, seem to improve the performance on the languages also containing these feature. It implies that Transformer models learn these language-specific properties. It is true for both mBERT and RoBERTa.
In the context of cross-lingual NER, we need to keep in mind that while we run our experiments on many languages, the feature space considered is relatively small. A side-effect is that when proving a system to be language-dependent in our feature experiment, we can only explain it with features at hand. However, countless other features are inevitably impacting the results; many of them might not even be described by any system yet (even an enormous WALS).

8.2 Contribution

1. We used Wikiann datasets for our experiments (before CoNLL was mainly used)

2. We divided languages into groups according to European Sprachbund (ES) features (see gitHub)

3. We demonstrated which features from ES might have an impact on cross-lingual learning with multilingual Transformers for NER

4. We confirmed findings from 'How multilingual is multilingual BERT?' and 'Do We

Need Word Order Information for Cross-lingual Sequence Labeling' - that Word Order matters (is incorporated in weights as well)

5. Additionally, we showed cross-lingual transfer ability to non-pretrained languages and across scripts in two case studies

6. We provided a classification of cross-lingual methods used in NLP (and for NER) and listed the work done so far on that topic during last (intense) few years.

8.3 Future Work

With our experiments we got one step further at developing a universal NER language system - we figured out which language-specific information might be incorporated in mBERT and XLM-R by our Features experiment. In further work, layers responsible for that features could be identified, removed, and the same experiments rerun to see if mBERT improves cross-lingually. Moreover, we confirmed using one model for several languages works better for similar languages. We encourage further work for more distant languages, especially in serving low-resource settings.

As a last point we want to say, that it is not reasonable to expect people to have dual degree, however, NLP start to be more and more challenging not only in the topic of growing number of models (a new state-of-the-art every month - hard to follow) but also with the advanced linguistic knowledge scientists need to optimise cross-lingual architectures or just to design reliable experiments. The gain is not just one-sided. NLP software also helps scientific typology filling feature databases with missing entries derived from known attributes.[**?**]

9 Addendum

While analysing our results, we came across a paper from Lauscher et al. (May 2020) - having a comparable experiment domain as us - and a few other very recent publications related to our research question. Obviously, they are not incorporated in this thesis but for the purpose of completeness we want to add a brief summary of their findings as an addendum.

Similar to us, Lauscher et al. evaluate Transformer models XLM-R and mBERT for NER on Wikiann dataset (zero-shot), but on a subset of 12 languages. Their fine-tuning centres on English and language representation vectors generated with LANG2VEC (based on linguistic features from URIEL database[**?**]).
They study language similarity, the impact of data size, the role of different tasks and ways to improving zero-short transfer for distant languages with a small supervision. Their results show as ours decreasing behaviour in the f1-scores when doing cross-lingual transfer from English to other languages in comparison to the monolingual scores of English.
The impact of syntactical, phonological, inventory features and similarity of the target language about family language, geographical and datasize properties show for NER a high correlation (Pearson) for phonological and geographical features (range 0.75 - 0.78). Nonetheless, also the rest four feature categories show an influence on the cross-lingual learning (correlation range 0.23 - 0.56, where the latter number belongs to syntactical features).
The language's datasize in the pre-training phase of mBERT (based on the size of wikipedia articles for each language from October 2018) present in opposite no visible correlation. Besides this, their few-shots results state out, just adding a few annotated sentences in the target language improves f1-scores significantly. For example NER-score across all target languages trained on English and on 10 languages improve by around 8 %. But the increase to more languages flattens and the increase with 1000 languages in comparison to just train it on English is around 14 %.[**?**] Our Feature experiments showed better f1-scores of fine-tuning on language groups as well even if we kept a zero-shot settings.
The publication in April 2020 from Groenwold underlines the findings our findings partially as well. By looking at the cross-lingual performance for text classification of eight models (inter alia mBERT) across languages and the morphological, syntactical and word order features derived from WALS database they share, they cannot clearly identify correlations with one of them but see a stronger performance of language pairs of the same language family.[**?**]

Furthermore, there seems to be a general recent interest in evaluating the multilingual ability of systems. For example, Hu et al (April 2020) published a multilingual evaluation tool, called XTREME, letting cross-lingual systems test their performance across 40 languages from 12 language families and 9 tasks in zero-shot conditions.[**?**] Another one is from Liang et al. (Apr 2020) called XGLUE following the same idea for 11 tasks and on two different multilingual datasets. [**?**]

Our research focused on the task Named Entity Recognition and from previous findings we could see that the influence of different typological properties between languages can have a different effect depending on the NLP task [**?**]. It seems sensible investigate how to overcome cross-lingual challenges task-specific before trying to develop a general system across different language understanding task with little cross-lingual biases [26].

[26] Also Artetxte et al. (Apr 2020) argues for this https://arxiv.org/pdf/2004.14958.pdf

List of Tables

List of Figures

10 Appendix

Dataset/Language	EN	NL	ES
Wikiann (full)	81281782	6060789	6060789
ConLL (train)	0.73	**0.89**	0.72
ConLL (test)	0.66	0.67	**0.86**

Table 8: No of lines for the datasets in preexperiments

References

[1] Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83, 2017.

[2] Emily M Bender. On achieving and evaluating language-independence in nlp. *Linguistic Issues in Language Technology*, 6(3):1–26, 2011.

[3] Shreenivas Bharadwaj, Vinayak Athavale, Monik Pamecha, Ameya Prabhu, and Manish Shrivastava. Towards deep learning in hindi ner: An approach to tackle the labelled data scarcity. In *arXiv:1610.09756*, 2016.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[5] Ryan Cotterell and Kevin Duh. Low-resource named entity recognition with cross-lingual, character-level neural conditional random fields. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 91–96, 2017.

[6] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*, 2014.

[7] Mónica Marrero, Julián Urbano, Sonia Sánchez-Cuadrado, Jorge Morato, and Juan Miguel Gómez-Berbís. Named entity recognition: fallacies, challenges and opportunities. *Computer Standards & Interfaces*, 35(5):482–489, 2013.

[8] Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2536–2545, 2017.

[9] Jian Ni, Georgiana Dinu, and Radu Florian. Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection. *arXiv:1707.02483*, 2017.

[10] Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, 2017.

[11] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[12] Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228, 2016.

[13] Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. Cross-lingual word clusters for direct transfer of linguistic structure. In *NAACL*, pages 477–487, 2012.

[14] Yonghui Wu, Min Jiang, Jun Xu, Degui Zhi, and Hua Xu. Clinical named entity recognition using deep learning models. In *AMIA Annual Symposium Proceedings*, volume 2017, page 1812. American Medical Informatics Association, 2017.

[15] Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A Smith, and Jaime Carbonell. Neural cross-lingual named entity recognition with minimal resources. *arXiv:1808.09861*, 2018.