Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
D-70569 Stuttgart

Master Thesis

Cross-Lingual Metaphor Detection for Low-Resource Languages

Anna Hülsing

Studiengang:	M.Sc. Computational Linguistics
Prüfer*innen:	Prof. Dr. Sabine Schulte im Walde
	Dr. Michael Roth
Beginn der Arbeit:	02.11.2022
Ende der Arbeit:	02.07.2023

Erklärung (Statement of Authorship)

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und dabei keine andere als die angegebene Literatur verwendet habe. Alle Zitate und sinngemäßen Entlehnungen sind als solche unter genauer Angabe der Quelle gekennzeichnet. Die eingereichte Arbeit ist weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen. Sie ist weder vollständig noch in Teilen bereits veröffentlicht. Die beigefügte elektronische Version stimmt mit dem Druckexemplar überein.[1]

(Anna Hülsing)

[1] Non-binding translation for convenience: This thesis is the result of my own independent work, and any material from work of others which is used either verbatim or indirectly in the text is credited to the author including details about the exact source in the text. This work has not been part of any other previous examination, neither completely nor in parts. It has neither completely nor partially been published before. The submitted electronic version is identical to this print version.

Für Simon und Max.

Abstract

State-of-the-art metaphor detection (MD) models achieve human-like performance for English data, while studies on MD for low-resource languages are currently missing. This thesis explores cross-lingual approaches that harness data from English, a high-resource language, in order to classify data in the target languages of Russian, German and Latin, either without using training data from the target languages or with as little as 20 instances. These instances were taken from the test data, but could also be created manually due to the small amount of annotating effort. The experiments indicate that the neural cross-lingual models mBERT (zero- and few-shot classification) and mBERT-based MAD-X perform well for German and Russian, while for languages where little data was used to pretrain mBERT, non-neural cross-lingual models with vector space model and conceptual features (abstractness, supersenses) outperform the mBERT-based models, if default hyperparameters are used.

No validation data in the target languages was available for performing hyperparameter-tuning. Therefore, as a byproduct it was discovered that, while using a source language dataset for validation leads to overfitting, using a dataset from another language rather than the source language leads to decent results. This is especially true for the MAD-X model, which – with the help of successful hyperparameter-tuning – outperforms the non-neural classifier for the low-resource language Latin.

Contents

1 Introduction

Song titles such as *Life is a Highway* are prominent examples of how we use metaphors in our everyday life. However, songs are by far not the only habitats of metaphors: on average metaphors are found in every third sentence across multiple domains (Shutova and Teufel, 2010). In cognitive linguistics, a (conceptual) metaphor is "defined as understanding one conceptual domain [A] in terms of another conceptual domain [B]" (Kövecses, 2010). In the above example, the domain *Life* (A) is understood in terms of the domain *Journey* (B). The conceptual metaphor *Life is a Journey* surfaces by means of a metaphorical linguistic expression. These are the words or expressions that "come from the language or terminology of the more concrete conceptual domain (i.e., domain B)" (Kövecses, 2010). The metaphorical linguistic expression in the example is *Highway.*[2]

Detecting whether or not an expression is a metaphorical linguistic expression (i.e. whether or not it is used metaphorically) is vital for many practical applications, such as sentiment analysis, machine translation, information extraction, dialog systems, and so on (Tsvetkov et al., 2014). Here, metaphor detection (MD)[3] can be one step of an NLP-pipeline.

Many efforts have been made to tackle the task of MD, and successfully so: close-to-human performance was seen in models using large pretrained language models like BERT (Devlin et al., 2019) for datasets containing single sentences with a metaphorical expression (Ma et al., 2021). Most MD experiments, however, are performed for English. Research on MD in other languages, especially in low-resource languages that are potentially typologically different from English, is missing, even though there would be several options of exploring MD for low-resource languages, such as data augmentation or cross-lingual knowledge transfer. **In this thesis, we explore the latter approach, namely how materials[4] from high-resource languages can be harnessed for detecting metaphors in low-resource languages**. This

[2]While this thesis focuses on metaphors, also the broader terms *figurative* and *non-literal* are used in the following sections. For more information on the terminology, see Appendix A.2.

[3]A list of abbreviations is found in Appendix A.1.

[4]Pretraining data, word ratings and annotated MD datasets, etc.

is done by investigating the following research questions:

1. Neural cross-lingual transfer methods have been shown to perform well on a wide range of tasks (such as named entity recognition and question answering). Are they also applicable to MD?
2. There are languages where only a small amount of pretraining data is available for large language models and/or that are typologically distant from the source language. Can a non-neural classifier outperform neural models for these languages?
3. It has been shown that sentences containing metaphorical language seem to be more emotionally charged than non-figurative sentences (Mohammad et al., 2016). Will adding emotion scores as conceptual features improve the performance of the non-neural classifier?
4. Research has shown that mid-range abstractness ratings are unreliable, since they exhibit a large degree of disagreement among annotators (Pollock, 2018). Will separating mid-range abstractness ratings from truly concrete and abstract ratings improve the performance of the non-neural classifier?

In order to answer these questions, the metaphor detection method (i.e. whether we use word-based or sentence-based MD, or sequential labelling), the transfer methods and the languages had to be chosen. As for **metaphor detection method**, we focus on word-based classification, since most datasets are created in such a way that the metaphoricity of one word – usually the verb – is annotated, as in the following example from the metaphor dataset by Tsvetkov et al. (2014):

(1) Miranda <u>opened</u> her mouth to reply, but no words came out. → *literal*

(2) Actions <u>talk</u> even louder than phrases. → *non-literal*

The binary classification task is to detect whether the underlined words, i.e. the target words, are used metaphorically in the given context or not. For the second example, sentence-based classification could also be carried out, but in the first example this is not possible: the combination *words came out* should be classified as non-literal, whereas *opened her mouth* should be classified as literal. Therefore, it is not possible to assign one label to the entire sentence. Most datasets are probably created for word-based MD, because many sentences in real life contain more than one expression that can be classified as literal or non-literal.

As for neural cross-lingual **transfer methods**, zero- and few-shot classification with multilingual BERT (mBERT, Devlin et al. (2019)) and the adaptation method MAD-X (Pfeiffer et al., 2020b) are used, as they have shown state-of-the-art results for other tasks. As a non-neural classifier, a random forest classifier using a vector space model and conceptual features (abstractness and supersenses) is used that is based on a classifier by Tsvetkov et al. (2014), since random forest classifiers perform well in low-resource scenarios and conceptual features in theory work well for every language.

language	**branch**	**# Wikipedia articles**
English (source)	Germanic	$\approx 6.6m$
German	Germanic	$\approx 2.7m$
Russian	Slavic	$\approx 1.9m$
Latin	Italic	$\approx 0.1m$

Table 1: Table presenting the target languages for the experiments (with English as the source language) for comparison with regard to language similarity and amount of pretraining data (m stands for millions).

As for **languages**, this thesis investigates how well the classifiers perform on the target languages German, Russian and Latin, because the amount of data used to pretrain mBERT varies greatly across these three languages and because they stem from different branches of the Indo-European language family (see Table 1)[5]. English as a high-resource language is the source language for the cross-lingual transfer.

[5]Numbers taken from `https://meta.wikimedia.org/wiki/List_of_Wikipedias`.

Whereas German and Russian are not considered low-resource languages in terms of pretraining data, they can be considered low-resource languages as we use little or no labelled training data in this thesis; the existing MD datasets are used for evaluation only. This way, insights gathered in this thesis can be used for languages where truly no training data is available. Latin, on the other hand, is a low resource language in terms of pretraining data and in terms of labelled training data. The different degrees to which each target language is considered low-resource are illustrated in Table 2.

language	**pretraining data**	**training data**
German	high resource	low resource (simulated)
Russian	high resource	low resource (simulated)
Latin	low-resource	low resource

Table 2: Table illustrating in what way each target language is considered a low-resource language. Whereas testing data could be gathered for German and Russian, we simulate that no training data is available.

The main contribution of this thesis is to identify which model performs best for each type of the given languages in terms of pretraining data and possibly language similarity between source and target language. This way, a suitable model can be picked for a downstream task in a comparable language. An additional contribution of this thesis is an insight into whether feature engineering in the areas of emotion and abstractness improves the performance of the non-neural classifier.
We begin by depicting related work in Section 2, which places this thesis into the broader scientific context. After that, the datasets for the source language English and the target languages Russian, German and Latin are presented, as well as the preprocessing steps that were necessary to make these datasets work for the different models (see Section 3). As a next step, the models – random forest classifier, zero-/few-shot classification with mBERT and MAD-X – are depicted in Section 4. In Section 5 the experiments that were carried out using the different models are presented along with the results for each experiment. The results are discussed in Sec-

tion 6. Qualitative analyses (see Section 7) try to answer questions that were raised during the discussion, while the conclusion (Section 8) summarizes the findings. As a last step, topics for future work (Section 9) are described. The code used for this thesis is accessible here: `https://github.com/AnHu2410/MD_crosslingual.git`.

2 Related Work

This section presents previous work on metaphor detection and cross-lingual representations in order to describe the scientific context of this thesis. The studies on metaphor detection form the basis for the methods that are used during the thesis, whereas cross-lingual representations that have so far been used for tasks like question answering are depicted here because in the thesis they are applied for the task of MD.

2.1 Metaphor Detection

Birke and Sarkar (2006) adapted a word-sense disambiguation approach to classify literal and non-literal usages of verbs by redefining literal and non-literal as two different senses of a word. They used seed sentences that were divided into a literal and a non-literal set and computed the similarity of a test sentence and the sentences from the two seed sets. A higher similarity score between the test sentence and the non-literal seed set indicated non-literal use (the opposite is true for literal use).
Turney et al. (2011) were the first to use insights from cognitive linguistics for their model, namely the insight that metaphors transfer knowledge from a concrete domain to an abstract domain. For the example *Life is a Highway*, knowledge from the domain *Journey* (involving concrete aspects like uphill and downhill slopes, roadbends and speed) is transferred to the abstract domain of *Life* by the metaphor. Since metaphoricity is therefore correlated with the degree of contextual abstractness, the authors used abstractness scores of the context words as features for a logistic regression model. This model performed on par or better than the model by Birke and Sarkar (2006), and was in contrast able to generalize to new words due

to the conceptual nature of its features.

The idea of "conceptual features" also inspired Tsvetkov et al. (2014), who used abstractness scores, imageability scores and semantic supersenses as classification features. The latter are semantic categories originating from WordNet (Fellbaum, 1998), such as *noun.body* or *verb.motion*. Consider the following sentence:

(3) The car drinks gasoline.

Here, they would use the feature $< verb.consumption, noun.substance >$ to represent *drink gasoline*, which indicates metaphorical use (while the feature $< verb.consumption, noun.food >$ would indicate literal use). The authors combined these conceptual semantic features with vector space word representations for a random forest classifier. Whereas Turney et al. (2011) focused on English data only, Tsvetkov et al. (2014) trained on English data and then evaluated the model on English, Spanish, Farsi and Russian. High evaluation scores showed that their model can be transferred to any other language (with the help of bilingual dictionaries). In the thesis, this model forms the basis for our non-neural MD system.

Köper and Schulte im Walde (2016) focused on MD of German particle verbs. Consider the following examples (particle verbs[6] in bold):

(4) Den Lippenstift solltest du **abschminken**. → *literal*

(5) Den Job kannst du dir **abschminken**. → *non-literal*

They also used conceptual semantic features for a random forest classifier: they used (1) abstractness and imageability ratings as well as (2) scores indicating the distributional fit of particle verbs / base verbs and the context. For example, they computed the distributional similarity between the base verb (*schminken*) and the object of the particle verb (*Job*/*Lippenstift*). Whereas the distributional similarity between the base verb and *Lippenstift* is rather high – this indicates literal usage –, the similarity between *schminken* and *Job* is rather low, which indicates non-literal

[6]A particle verb consists of a base verb (in this case: *schminken*) and a particle (*ab*-).

usage. The similarity scores were used as features for the classifier. In addition, they used (3) unigrams (unigram features consisted of the output of a classifier (literal/non-literal) that takes unigrams as input) and (4) noun clusters (nouns in the PV sentence are replaced with their corresponding cluster tag from unsupervised learning) as features. These four feature types led to state-of-the-art performance in 2016, underlining the importance of conceptual features.
Do Dinh et al. (2018) were the first to use a neural model architecture for MD. They used an LSTM model to encode the left and right context of the target word[7] and a dense layer for the target word itself. Both the context and the target words were represented by pre-trained word-embeddings. The outputs of the two LSTMs and the dense layer of the target word were then concatenated and fed through more dense layers before the softmax layer. Their approach performed better than or comparable to existing models; however, no feature engineering was necessary.
Based on previous research in linguistics and psychology indicating that metaphorical phrases tend to be more emotional, Dankers et al. (2019) explored the relationship between metaphors and emotions by building several multi-task learning models. MD was modelled as a binary task, whereas emotion prediction was a regression task predicting valence, arousal and dominance scores (i.e. scores of the three-dimensional VAD emotion model). They used various multitask learning architectures to predict metaphoricity and VAD-scores jointly. The best performing architecture made use of BERT embeddings, which were not fine-tuned but instead used as input to a multilayer perceptron or to additional attention layers. They reached state-of-the-art results in 2019 for both metaphor and emotion prediction.
Ma et al. (2021) simplified the architecture by fine-tuning BERT for MD. To perform word-based binary metaphor classification, they masked the target word and concatenated the original sentence twice – once with the masked target word, once without masking anything. BERT then predicts whether the two sentences appear in the same context; if BERT predicts both sentences to be in the same context, then it is probable that the masked word is meant literally, otherwise it is meant metaphor-

[7]The target word is the word for which it should be decided whether it is used metaphorically or not.

ically. They also performed sentence-level classification and sequential labelling of metaphorical expressions. Their results showed a huge increase over previous state-of-the-art models with close-to-human performance. Their word-based classification approach is used for the mBERT-based classifiers in the thesis. However, their approach was carried out only for English and not in a cross-lingual setting.
Frassinelli and Schulte im Walde (2019) explored the pattern of abstract words cooccurring with abstract words and concrete words cooccurring with other concrete words. They found that the pattern concrete verbs – concrete context and abstract verbs – abstract context is generally confirmed for verb-noun-subcategorisation. However, in the case of non-literal language, this pattern did not emerge. They therefore confirmed that abstractness can be a useful feature for MD. For their study they only used words that have a very high or very low concreteness rating, since it has been shown, for example by Pollock (2018) that words with mid-range ratings are hard to categorize unambiguously. In the thesis, the non-neural classifier also treats mid-range ratings separately.

2.2 Cross-Lingual Respresentations

Vulić and Moens (2013) proposed a bootstrapping method to create bilingual vector spaces from non-parallel data. Usually, a high-dimensional vector in a feature vector space uses context features as dimensions. For the proposed bilingual vector space, these features consisted of translation pairs (for example, the English-German translation pair $< dog, Hund >$). To initialize the vector space, the authors relied on multilingual probabilistic topic modelling (MuPTM), which yielded confident translation pairs for high frequency words. From this method, symmetric translation pairs were obtained and ranked by frequency. Only the most frequent pairs were used as seed words for the vector space.
As a next step, bootstrapping was performed by (1) building a context vector for each source and target word, (2) computing a similarity score between each source word and all target words (this is possible because source and target words live in the same vector space), (3) picking the most promising translation candidate (for

example, by comparing cosine similarity scores) and (4) adding these highly reliable translation pairs as new dimensions. Steps (1) to (4) were then repeated until convergence, so that more and more new dimensions were added to the seed words. This method can be applied to any language pair. It is suitable to produce bilingual vector space representations for features in a cross-lingual MD model.

Lauscher et al. (2020) showed the limitations of large multilingual pretrained language models, such as mBERT, that are currently used as a default in cross-lingual NLP-tasks. They indicated that these models do not transfer knowledge well for low-resource target languages (i.e. languages with small pretraining corpora) and for distant language pairs. They investigated low-level tasks (part of speech tagging, dependency parsing and named entity recognition) as well as high-level tasks (natural language inference and question answering). They also showed that fine-tuning on large amounts of e.g. English data first and then continuing fine-tuning with very few examples from the target language considerably improves results across all languages and tasks. The thesis investigates whether these findings also apply for MD. In addition, they found that the performance of a model can be predicted for the given tasks by taking into account the amount of pretraining data and the distance between source and target language.

Pfeiffer et al. (2020b) tried to mitigate the problems multilingual language models have with low-resource languages (or languages unseen during pretraining) by using an adaptation method, i.e. by inserting small amounts of trainable weights into an existing pretrained model. For their method, the authors proposed three types of adapters: language adapters, task adapters and invertible adapters. Firstly, they froze the pretrained model and trained one language adapter on a masked language modelling task in the source and one in the target language on unlabelled data. As a second step, they plugged in only the source language adapter and injected the task adapter and trained it on labelled data from the source language. As a third step, they performed inference after plugging in the target language adapter and the (language-agnostic) task adapter.

The invertible adapters were plugged in simultaneously with the language adapters, but had a slightly different architecture, because they adapted the embeddings,

while the language and task adapters were inserted into each transformer layer. This method is called Multiple ADapters for Cross-lingual transfer (MAD-X) and is applied for MD in the thesis.
Ansell et al. (2022) combined sparse fine-tuning with a modular approach similar to MAD-X. Based on the Lottery Ticket Hypothesis, the authors fine-tuned a pre-trained language model on a task or a language. Then, the parameters that showed a small absolute difference to their value before fine-tuning were masked out. The model was reset to its original values, but only those parameters that had not been masked out were trainable in the next step. This process of sparse fine-tuning (SFT) was repeated to find the "winning tickets", i.e. the subnetworks that led to good results for a given task or language.
SFT was carried out for the source and target language first. In order to perform the SFT for the task, the source language SFT-representation was applied. For inference, the task SFT-representation was combined with the target language SFT-representation (cf. MAD-X) for zero-shot classification. They applied this method to languages seen and unseen during pretraining for the following tasks: part-of-speech tagging, dependency parsing, named entity recognition and natural language inference. In future work, it would be worth trying out this approach for MD, since it seems to work well even on languages unseen during pretraining.

3 Datasets and Preprocessing

The following sections describe the datasets that were used for the languages English, Russian, German and Latin in the experiments. They also show how each dataset was preprocessed. Since the random forest classifier and the mBERT-based classifiers require different outcomes of the preprocessing step, each language in this section is depicted with two preprocessing methods: The **random forest classifier** used in the experiments extracts features from the target word, i.e. the verb to be classified, and its subject and object (if available). Therefore, as a preprocessing step, verb, subject and object had to be extracted from the instances and lemmatized. Here is an example for English:

(6) The bus eventually arrived.

In this case, subject and verb have to be extracted and lemmatized, so the goal of the preprocessing is to find the two lemmata *bus* and *arrive*. For further details on feature extraction from subject, verb and object and on the random forest classifier in general, see Section 4.1.
For the **mBERT-based classifiers**, the data had to be preprocessed in such a way that the target word is masked in a copy of the original sentence. For example sentence (6), the goal of the preprocessing was to obtain the following sentence:

(7) The bus eventually [MASK].

The masked sentence and the original sentence then had to be further preprocessed by the HuggingFace tokenizer pipeline[8]. For further information on how the mBERT-based models use this input for classification see Sections 4.2 and 4.3. Whereas this section focusses on generating the input for the classifiers, Section 4 provides a more detailled description of the different classification models.

3.1 English

3.1.1 Dataset

As to training data for the source language English, we used the metaphor detection dataset[9] provided by Tsvetkov et al. (2014). It was collected from the TenTen Web Corpus, which contains “linguistically valuable web content”[10]. It consists of 222 sentences, of which 111 are annotated as metaphorical and 111 are annotated as literal, so the dataset is balanced. To be more specific, for each sentence it was annotated whether the target verb is used metaphorically or literally:

(8) Her son **broke** my window. → *literal*

[8]https://huggingface.co/docs/transformers/main_classes/tokenizer
[9]https://homes.cs.washington.edu/~yuliats/#publications
[10]https://www.sketchengine.eu/ententen-english-corpus/

(9) My computer battery **died**. → *non-literal*

In addition, the subject, verb and object are given as lemmatized forms. 93 of the sentences contain both subject and object, 94 contain a subject only and 35 contain an object only. There are no sentences where both subject and object are missing, and all subject and objects consist of nouns. The mean sentence length is 11 words with a standard deviation of 3.8 words.[11]

3.1.2 Preprocessing

As the subjects, verbs and objects were already given in the dataset, no preprocessing was necessary for the **random forest classifier**. For the **mBERT-based classifiers**, we masked the target verb, as one can see in the following sentence:

(10) The twentieth century saw intensive development of new technologies.

However, in the dataset only the lemmata of the target verbs (here: *see*) were given and not the inflected forms (here: *saw*) that occur in the sentences and which had to be masked. Therefore, we tokenized the sentences first and then lemmatized all tokens using the WordNetLemmatizer from the NLTK library (Bird et al., 2009). Since this step produced errors (in the example, the token *saw* was lemmatized as *saw* instead of *see*), we used a heuristic: Firstly, we checked whether the first two letters of the lemmatized form (*se* in the example) also appear as the first two letters in one of the sentence tokens. In some cases, this led to finding the correct token. For words like *saw*, this procedure did not lead to a result (since *se* is not identical with *sa*, the correct token was not selected). Therefore, a second round of comparison

[11]Tsvetkov et al. (2014) used a different dataset as their training set, namely a filtered version of the TroFi Example Base (Birke and Sarkar, 2006); the dataset that we used for training was used by them as their test set. Even though we want to reproduce their results, we do not use the same training dataset as they did, because in the TroFi Example Base subject, verbs and objects are not marked, and it would be too time-consuming to add this information. As we do not know exactly which instances they selected in their filtering process, the training dataset would not be fully comparable anyway.

was carried out where we checked whether the first letter of the lemmatized form (s) was identical to the first letter of one of the tokens (s). In the case of *see* this produced the correct token. Since especially the last step is error-prone, we checked the final result manually (no errors occured for our data).
This procedure created two inputs for the classifier, namely the original sentence (10) and a copy of the original sentence, where the target word is masked (11):

(11) The twentieth century [MASK] intensive development of new technologies.

However, the mBERT-based models require further preprocessing of these two input strings, which is done automatically by the HuggingFace tokenizer pipeline[12]:

- As a first step, the input string is split into subword token strings using WordPiece (for a description of the algorithm see Appendix A.4). This is an example for the subword tokenization of the sentence "My computer slipped into coma.", which is found in the English dataset:

  (12) 'My', 'computer', 'sl', '##ip', '##ped', 'into', 'coma', '.'

- Also, special tokens are added, such as the token that indicates the beginning of a sentence ([CLS]) or the token that indicates where one input string ends ([SEP]). For tasks such as masked language modelling, the mask token would be added at this point, too. This is not necessary in our case, as the target word was already masked with the mask token during the previous preprocessing steps. Adding special tokens leads to the following representation:

  (13) [CLS] My computer slipped into coma. [SEP] My computer [MASK] into coma. [SEP]

- Each of the subtokens and special tokens is mapped to a numerical ID:

[12] `https://huggingface.co/docs/transformers/main_classes/tokenizer`

(14) [101, 11590, 18765, 38523, 17437, 16898, 10708, 18452, 119, 102, 11590, 18765, 103, 10708, 18452, 119, 102]

Besides these input-IDs, the tokenizer also returns an attention mask and token-type-IDs. The attention mask indicates which input-IDs to pay attention to: in the case of padding, some input-IDs do not contain actual information, since their purpose is to create inputs of identical length. To these input-IDs no attention is paid. The token-type-IDs indicate where the first sequence (in our case the first, unmasked sentence) ends and where the second sequence (in our case the masked version of the original sentence) begins. Input-IDs, the attention mask and token-type-IDs are the outcome of the preprocessing and are fed to the mBERT-based classifiers.

3.2 Russian

3.2.1 Dataset

For Russian, we used the metaphor detection dataset[13] provided by Tsvetkov et al. (2014). It consists of 240 sentences, of which 120 are annotated as metaphorical and 120 are annotated as literal, so the dataset is balanced. It is based on the ruTenTen-corpus[14]. Here are two example sentences:

(15) Бедность давит на людей. (translation: “Poverty weighs on people.”) $\rightarrow$ *non-literal*

(16) Повар варит суп на кухне. (translation: “The cook cooks soup in the kitchen.”) $\rightarrow$ *literal*

Subject, verb and object are given as lemmatized forms. 113 of the sentences contain both subject and object, there are 77 sentences that contain a subject only and 50

[13]`https://homes.cs.washington.edu/~yuliats/#publications`

[14]`https://www.sketchengine.eu/rutenten-russian-corpus`

sentences that contain an object only. There are no sentences where both subject and object are missing. The mean sentence length is 9 words with a standard deviation of 3 words.

3.2.2 Preprocessing

As the dataset includes subjects, verbs and objects as lemmatized forms, no preprocessing was necessary for the **random forest classifier**. For the **mBERT-based classifiers**, we carried out the same preprocessing that was described for the English dataset in Section 3.1 (i.e. we replaced the target word with the mask-token and preprocessed the original sentence and its masked copy with the HuggingFace tokenizer pipeline).

3.3 German

3.3.1 Dataset

For German, we used the metaphor detection dataset provided by Köper and Schulte im Walde (2016), which is publicly available[15]. It contains 6436 sentences; 4174 of these are annotated as literal and 2262 are annotated as non-literal. In addition, the dataset contains syntactic information from dependency parsing for each word. The dataset also indicates which token in the sentence is the target word and all words in the sentence are given in the lemmatized form. The sentences were originally taken from DECOW14AX, which is a German web corpus (Schäfer and Bildhauer, 2012). Here are two example sentences from the dataset:

(17) Wer nun nicht vom Glauben abfällt dem ist wirklich nicht mehr zu helfen! [sic] (translation: "Those who do not lose their faith now cannot be helped.") $\rightarrow$ *non-literal*

[15]`https://www.ims.uni-stuttgart.de/forschung/ressourcen/experiment-daten/pv-nonlit/`

(18) Den Panzern fallen die Schussrohre ab. (translation: “The tanks are losing their shooting pipes.”) $\rightarrow$ *literal*

To make the dataset more comparable to the Russian dataset, the number of literal instances was reduced so that the dataset is balanced (896 metaphorical sentences and 896 literal sentences). In this version, 307 of the sentences contain both subject and object, 471 contain a subject only and 765 contain an object only. There are 249 sentences where both subject and object are missing. The mean sentence length is 13 words with a standard deviation of 3.5 words.

3.3.2 Preprocessing

For the **random forest model**, knowing the subject and object pertaining to the target word is necessary. The target word is indicated, but its subject and object are not given explicitly in the dataset. The dataset contains information from dependency parsing, namely the head of each word (which can be another word or the root token) and the syntactic functions of most words. There are, however, several problems in using this information to find the subject and object of the target word:

- Passive: Many sentences are written in the passive voice. As this is not the case for the English dataset, the subjects of passive sentences should become the objects. Here is an example:

  (19) Es wurden dreimal täglich die Mahlzeiten aufgetischt. (translation: “Meals were served three times a day.”)

  In this sentence the word *Mahlzeiten* is parsed as the subject of *aufgetischt*. However, in active sentences *Mahlzeit* and related words seldom are the subject of the target verb *auftischen* (except for figurative language), hence a classifier trained on English (non-passive) data will not be able to classify it correctly. Subjects in passive sentences were therefore converted to objects.

- Non-standard language: The dataset contains non-standard language, which is not parsed correctly, as in the following example:

  (20) Wahnsinn, unvergesslich, kein Oscar für diese Filmmusik, dafür **gehört** jede Academy der Welt eingerissen. (translation: “This is madness, unforgettable, no oscar for this soundtrack, for this every Academy in the world should be destroyed.”)).

  In cases like this one, the correct subject and object had to be annotated manually.

- Coreference resolution: In cases, where the subject or object of an target word was a relative pronoun, the corresponding noun from the main clause was used as subject or object respectively. (There are, however, many pronouns that are not resolved, for example demonstrative pronouns, and which therefore do not yield much information for feature extraction.)

Therefore, we used the syntactic dependencies found in the dataset to find the subjects and objects, but also carried out a manual correction as described to find the lemmatized subjects and objects for passive voice, non-standard language and coreference resolution.
For the **mBERT-based models**, the sentences were masked. As the target word was marked in the dataset, we were easily able to create the mask. We then further preprocessed the original sentence and its copy with the masked target word by using the tokenization pipeline described in Section 3.1 for English, which produced the final input to the mBERT-based classifiers, namely input-IDs, the attention mask and token-type-IDs.

3.4 Latin

3.4.1 Dataset

As the basis for the Latin dataset we used the Lexham Figurative Language of the New Testament Dataset by Westbury et al. (2016), which is published in the Logos Bible Software[16]. The authors annotated conceptual metaphors, "where each figurative expression is given a figurative category, source and target terms that represent the figurative concept (source) and intended meaning (target), and a brief description"[17]. The software itself shows the passages from the New Testament (the American Standard Version of the Bible was used), and highlights the metaphors in each verse. Using these highlights, we extracted 100 sentences, of which 50 were annotated as metaphorical and 50 were not. As the metaphors were annotated in the English Bible text, we then manually searched for the corresponding Latin translations in the Vulgate[18]. Here are two examples:

(21) Et venerunt, et impleverunt ambas naviculas, ita ut pene mergerentur. (text from American Standard Version from Logos: "And they came, and filled both the boats, so that they began to sink.") $\rightarrow$ *literal*

(22) Et dixerunt ei: Quia heri hora septima reliquit eum febris. (text from American Standard Version from Logos: "They said therefore unto him, Yesterday at the seventh hour the fever left him.") $\rightarrow$ *non-literal*

13 of the selected sentences contain both subject and object, 27 contain a subject only and 29 contain an object only. There are 31 sentences where both subject and object are missing. The mean sentence length is 22 words with a standard deviation of 7.5 words.
When selecting the literal and non-literal instances from the Logos Bible Software, two problems arose:

[16]`https://www.logos.com`
[17]`https://www.logos.com/product/178518/lexham-figurative-language-of-the-bible-glossary`
[18]`https://vulgata.info/index.php?title=Kategorie:BIBLIA_SACRA.`

- The annotation is not complete. Some instances are annotated as non-literal in one case, but a very similar wording is annotated as literal. For example, the verb *filled* is annotated as figurative in the following example:

  (23) And his father Zacharias was **filled with the Holy Spirit**, and prophesied, saying ...

  However, it is not annotated as figurative here:

  (24) ... and Elisabeth was **filled with the Holy Spirit**.

  In such cases, we assumed that the annotation was simply forgotten in the second text.

- Many instances were annotated as non-literal, but it is not obvious to theological laypersons, why these instances were considered non-literal. For example, the verb *followed* was annotated as non-literal in many cases, as it is used in the sense of "to be a disciple". We did not use these annotations in the dataset, because classifying them correctly would require expert knowledge, which is in stark contrast to the other datasets and which cannot be expected from the classifiers.

3.4.2 Preprocessing

For the **random forest classifier**, we manually annotated subject, verb and object and lemmatized them. For the **mBERT-based classifiers**, we selected the target word manually and masked it in the input string. For the original input string and the masked input string, we generated input-IDs, the attention mask and token-type-IDs as described in Section 3.1 for English.

4 Models

After the preprocessing, the data for all languages contained subject, verb and object (as lemmata) as input for the random forest classifier and the original sentence plus a copy of the original sentence, where the target word was masked, for the mBERT-based models. The following sections describe how the different classifiers use this input to make predictions, namely the random forest classifier (Section 4.1) and the mBERT-based models, which are zero- and few-shot classification with mBERT (Section 4.2) and an adaptation method based on mBERT, MAD-X (Section 4.3).

4.1 Random Forest Classifier with Vector Space Model (VSM)

Firstly, this thesis explores a non-neural classifier, namely a random forest classifier featuring a vector space model, which is based on the classifier by Tsvetkov et al. (2014). The features for this classifier are extracted from (lemmatized) subject-verb-object-triples of the corresponding sentences, where the verb is the target word to be classified.

4.1.1 Random Forest Classifier

Random forest classification is a classification method making use of ensembles of different decision trees. A **decision tree** consists of a root node, internal nodes and the leaves. Each node "splits the instance space into two or more sub-spaces" (Rokach and Maimon, 2014) according to a certain attribute. Each leaf corresponds to a class label that can be assigned to an instance.
A tree is created by first instantiating a root node. Then,

1. all features are compared for the node via a split criterion which calculates the impurity of the split (see below),
2. the feature with the best impurity score is chosen as the label for this node (as a running example we introduce the feature "abstractness_subject") and

3. as many edges are created as there are values connected to the feature (in the example, there could be three edges: “abstract”, “neutral”, “concrete”).
4. As a next step, a new subtree node is instantiated for each of the edges (i.e. three nodes for “abstractness_subject”).

This process is repeated for each node until a stopping criterion is fullfilled (Rokach and Maimon, 2014).
Two concepts need further explanation, namely the stopping criterion and the split criterion. For the implementation of the random forest classifier, we use the *scikit learn* library[19]. We use the default hyperparameters, where the stopping criterion is that “nodes are expanded until all leaves are pure”[19]. As default split criterion, we use the default *scikitlearn* criterion, called Gini index[20]. For each edge, a kind of leaf is simulated, i.e. a node that is not expanded further and where it is counted how many instances belong to each of the classes. This simulated leaf is exemplified with the green boxes in Figure 1, and for each of these leaves the following Gini impurity $H(Q_m)$ is calculated[20]:

$$H(Q_m) = \sum_k p_{mk}(1 - p_{mk}), \tag{1}$$

where Q_m is the subset of the data at this simulated node m (for example the 3 literal and 1 non-literal sentences in the first green box) and k is a class label (in the example, “literal” or “non-literal”). p_{mk} is the proportion of instances of class k at node m, and is defined as[20]:

$$p_{mk} = \frac{1}{n_m} \sum_{y \in Q_m} I(y = k) \tag{2}$$

In the example (Figure 1), for the first green box p_{mk} with $k = \textit{literal}$ would be $\frac{3}{4}$, with $k = \textit{non-literal}$ it would be $\frac{1}{4}$. The Gini impurity $H(Q_m)$ for the first green box would be *0.375* ($\frac{3}{4}(1-\frac{3}{4}) + \frac{1}{4}(1-\frac{1}{4}) = 0.375$). For the second green box, $H(Q_m) = 0$

[19]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[20]https://scikit-learn.org/stable/modules/tree.html#tree-mathematical-formulation

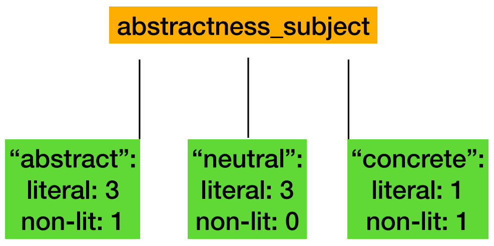

abstractness_subject
“abstract”: literal: 3 non-lit: 1
“neutral”: literal: 3 non-lit: 0
“concrete”: literal: 1 non-lit: 1

Figure 1: Example of calculating the Gini-index for a given feature.

and for the third green box $H(Q_m) = 0.5$. A Gini impurity of 0.5 signifies a maximally impure feature, whereas a Gini index of 0 indicates a pure feature, i.e. a feature where all instances that show this feature belong to one class.
The total Gini impurity G is calculated as the weighted sum of all simulated leaves. For example, $H(Q_m)$ for the first green box would be weighted by $\frac{4}{9}$, since it covers 4 instances of the total of 9 instances across all three green boxes. The total Gini index would in this example amount to a rounded value of 0.28. This value would be compared to the total Gini index for other features, such as “abstractness_object” or “valence_verb” and the feature with the lowest impurity would be chosen for the orange top node in the example. For inference, the test instance follows the path that corresponds to its features from the top node to the leaf, which assigns a label to the instance.
The **random forest classifier** used in this thesis instantiates 100 of these decision trees (in the experiments with default hyperparameters), which are trained on random subsets of the training data and where each decision tree considers a random subset of features. For inference, the test data is being classified with each individual tree and the results are averaged to make a classification decision.

4.1.2 Feature Types

In the paper by Tsvetkov et al. (2014), three categories of features are used for the random forest classifier:

- **Abstractness and Imageability**: Tsvetkov et al. (2014) use abstractness and imageability scores, which they generated on the basis of the MRC ratings created by Wilson (1988). They used a "logistic regression classifier to propagate abstractness and imageability scores from MRC ratings to all words for which [they] have vector space representations" (Tsvetkov et al., 2014). We also use these scores, which are published in their github repository[21].

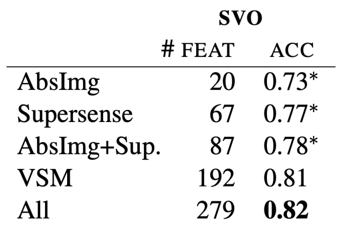
- **Supersenses**: Tsvetkov et al. (2014) refer to supersenses as "coarse semantic categories"" and use them as features in the following way: A word can belong to several synsets in WordNet (Fellbaum, 1998), each of which is associated with several supersenses. A feature vector is created with these supersenses as dimensions. For example: the noun *head* participates in 33 synsets, 3 of those are related to the supersense *noun.body*. The dimension corresponding to the supersense *noun.body* therefore gains the value of 3/33. The supersense scores for each word can be accessed on github[21].

- **VSM**: Tsvetkov et al. (2014) use the VSM by Faruqui and Dyer (2014). This model uses multilingual information in order to produce similar vectors for synonymous words. The scores can be accessed on github[21].

4.1.3 Feature Extraction

Features are extracted from the training data in two ways: Firstly, as individual features, and secondly, as combinations of features. For the individual features, the features of each feature type (see Section 4.1.2) are extracted from the lemmatized subject, verb and object. These three feature vectors are concatenated.

[21]`https://github.com/ytsvetko/metaphor`



	SVO	
	# FEAT	ACC
AbsImg	20	0.73*
Supersense	67	0.77*
AbsImg+Sup.	87	0.78*
VSM	192	0.81
All	279	**0.82**

Figure 2: Excerpt from Table 2 in Tsvetkov et al. (2014), which shows the number of features for each feature type (abstractness/imageability, supersenses, vsm and combinations of these types) which they used for their experiments.

For the combinations of features, features for token pairs (subject-verb, verb-object) are extracted. Tsvetkov et al. (2014) provide the following example: "to generate the feature vector for the SVO triple $< car, drink, gasoline >$, we compute all the features for the individual words *car*, *drink*, *gasoline* and combine them with the conjunction features for the pairs $car, drink$ and $drink, gasoline$". Here, they mention that they create combinations of features for "all" features. However, in the code that they provide[22], the combinations of features are only created for abstractness and imageability features, but not for the vsm and supersense features. Also, the number of features shown in Table 2 in Tsvetkov et al. (2014) shows that combinations of features are only created for abstractness and imageability (see the table in Figure 2). For the feature type abstractness and imageability, there are

- 3 individual features for abstractness: subject_abstractness, verb_abstractness, object_abstractness,
- 8 combinations of features for abstractness: subject_verb_aa, verb_object_aa, subject_verb_ac, verb_object_ac, subject_verb_ca, verb_object_ca, subject_verb_cc, verb_object_cc ("aa" stands for "abstract-abstract", "ac" stands for "abstract-concrete", etc.),
- 11 features for imageability analogous to the abstractness features.

[22] `https://github.com/ytsvetko/metaphor`

This list assumes 22 features, whereas 20 are mentioned in the paper (see 2). The difference is probably due to the fact that Tsvetkov et al. (2014) use a different training set, namely a downsized and preprocessed version of the TroFi Example Base by Birke and Sarkar (2006). Since this version is not publicly available, it can only be assumed that not all combinations of abstract and concrete values that were found in our English dataset were found in their training set.

It is clear, though, that the number 20 can only be achieved by using combinations of features in addition to individual features. This is not true for the other feature types: 67 features are listed for supersenses. As there are 26 supersenses for nouns, hence 26 supersenses for the subject and 26 for the object, as well as 15 supersenses for verb, the number 67 is possible only if no combinations of features are used. Since the vsm model contains 64 dimensions, the number 192 is also only possible without combinations of features, since the vectors for the three tokens subject, verb and object are concatenated ($64 \times 3 = 192$).

In the experiments in this thesis the 22 features listed above are used for abstractness and imageability. For the other feature types, no combinations of features are used. The numbers of features for each feature type used in our experiments are summarized in Table 3.

feature type	**# features**
Abstractness/Imageability	22
Supersenses	67
VSM	192
All	281

Table 3: Table presenting the numbers of features used in the experiments in this thesis for each feature type when reproducing the results from Tsvetkov et al. (2014).

4.1.4 Cross-Lingual Inference

Section 4.1.3 describes how features are extracted for training on the source language dataset. If inference was done in the source language, the features would be extracted in the same way from the test instances as described above for the training instances. However, to use the classifier cross-lingually, the features from the test instances are extracted in the following way:

- Subject, verb and object of the sentence from the target language are translated to the source language with the help of an electronic bilingual dictionary, which yields one-to-many translations (e.g. target word: *Fliege*, translation to source language: "fly, bowtie"). For Russian and German, the Word2Word library by Choe et al. (2020) was used for this purpose[23]. For Latin the translations were collected manually from an online Latin-English dictionary[24]. For a short discussion on the advantages and disadvantages of different approaches to obtaining one-to-many translations see Appendix A.5.

- The mean scores for all translations are calculated. In the case of abstractness, imageability and vsm scores, this is done conventionally by summing up the scores for the individual translations and dividing this sum by the number of translations.

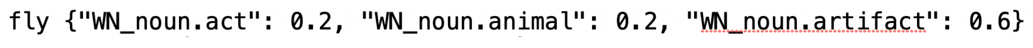
- For supersenses, averaging is done in a special way: Firstly, all synsets for the translations, e.g. "fly" (5 synsets) and "bowtie" (1 synset), are collected. Then it is counted how many synsets are associated with a certain supersense, for example with $noun.artifact$. In the case of *Fliege*, 1 synset is connected to this supersense for "bowtie" and 3 for "fly", so 4 overall. Lastly, the proportion of synsets associated with this supersense is calculated, which is to be 4/6, since in total 4 out of 6 synsets are connected to this supersense. Thus, in the supersense feature vector the dimension $noun.artifact$ gets the score 4/6.
  As the supersense scores for individual words were given in the github repository[25] by Tsvetkov et al. (2014), for this thesis a shortcut was taken: We gathered the number of synsets for each translation from NLTK (Bird et al.,

[23]Choe et al. (2020) extract bilingual lexica from parallel subtitle corpora. In doing so, they aim at high-coverage dictionaries, which suits our purpose well.

[24]`https://www.online-latin-dictionary.com`

[25]`https://github.com/ytsvetko/metaphor`

2009), namely that there are 5 synsets for “fly” and 1 synset for “bowtie”. Then the score by Tsvetkov told us that the supersense *noun.artifact* is connected to 60% of the synsets for “fly” (see Figure 3) and to 100% of the synsets for “bowtie” (not shown here). 60% of the 5 synsets means that 3 synsets are connected to this supersense. For “bowtie” there is only one synset and this was associated to the supersense *noun.artifact* (100%). By doing these caculations we know that in total 4 (3 from “fly” and 1 from “bowtie”) out of 6 synsets are connected to the supersense *noun.artifact* without having to download and iterate through the different lexicographer files from WordNet (Fellbaum, 1998), where each synset is assigned to one supersense.



```
fly {"WN_noun.act": 0.2, "WN_noun.animal": 0.2, "WN_noun.artifact": 0.6}
```

Figure 3: Example output for the noun “fly” from the supersenses-file from Tsvetkov et al. (2014).

Once the subjects, verbs and objects of the test sentences are translated and the scores are averaged, the concatenated vectors for all feature types are fed to the random forest classifier for inference.

4.2 Zero- and Few-Shot Classification with mBERT

One common way of performing neural cross-lingual transfer is to use multilingual pretrained language models, namely mBERT. BERT (Devlin et al., 2019) is short for Bidirectional Encoder Representations from Transformers. This means that it is an attention-based architecture (using *transformers*, see Vaswani et al. (2017)) that *bidirectionally encodes* large amounts of unlabelled data, in order to create contextual *representations* of an input sequence. The implementation of BERT consists of two steps: pretraining and fine-tuning.
During pre-training, the model is trained on unlabelled training data, which – in the case of mBERT – are the Wikipedias in 104 different languages[26]. For that, two

[26]The languages were chosen because they have 104 of the world’s largest Wikipedias, see `https://huggingface.co/bert-base-multilingual-cased`

tasks are employed, masked language modelling (MLM) and next sentence prediction (NSP). For MLM, a percentage of the input tokens (tokenized by Word Piece, see Appendix A.4) is masked, and the model has to predict the masked words. For NSP, a corpus is generated where in 50% of all cases sentence A is followed by its actual next sentence B, and in 50% of all cases it is followed by a random sentence B from the corpus. The model then has to predict whether sentence B is the next sentence or not. Since the labels (i.e. the words to be predicted in the case of MLM and the actual next sentence for NSP) for both pretraining tasks are found in the corpus itself, the tasks are carried out in an unsupervised way. This means that huge amounts of text can be used as training data: no labelling effort is necessary.
For fine-tuning, a classification layer is added to the pretrained model. Then all parameters – the pretrained ones and the classification layer – are fine-tuned on a downstream task.
In this thesis, this model is used in two scenarios, a zero-shot and a few-shot classification. For **zero-shot learning**, we will investigate how mBERT performs when it is fine-tuned on source language data and then evaluated on target language data. For **few-shot learning**, we will investigate whether double fine-tuning (first on source language data and then fine-tuning again on a few target language instances) boosts the classifier's performance, as was shown by Lauscher et al. (2020) for other tasks like POS-tagging and natural language inference.
In both scenarios, we fine-tune by adapting the procedure from Ma et al. (2021). We take the original sentence (sequence A), copy the sentence and mask the target word (sequence B), and then exploit the fact that mBERT was pretrained on the NSP task. If mBERT predicts that both sequences appear in the same context, then it is likely that the masked word is used literally. If mBERT predicts that the two sequences do not appear in the same context, then it is likey that the masked word is used non-literally. Ma et al. (2021) do not explain in detail why this method works. The following can be assumed, though: Shutova and Teufel (2010) show that according to their data, metaphors occur on average in every third sentence across various domains. This means that in two thirds of all sentences, so in the majority of cases, the words are used literally. Therefore the representation generated by mBERT for

the masked tokens tends to be on the literal side. Therefore, the masked sentences with literally used target words have a representation closer to the original than the non-literal sentences, and this similarity probably is the deciding factor in the classification decision.

4.3 Adapter-Based Crosslingual Transfer

Besides zero- and few-shot mBERT, a third neural method was implemented: MAD-X, which is short for Multiple ADapters for Cross-lingual transfer (Pfeiffer et al., 2020b). Here, small amounts of weights are injected into the embedding and transformer layers of pretrained multilingual models such as mBERT, while the pretrained model itself is frozen. Three types of adapters are used: language adapters, task adapters and invertible adapters.

Language and task adapters are added to each transformer layer. They take as input a hidden layer h_l. They first insert a matrix $D_{down} \in \mathbb{R}^{h \times d}$ to downsize h_l to a bottleneck dimension d, which is followed by a ReLU activation function and a multiplication with a matrix $D_{up} \in \mathbb{R}^{d \times h}$, which is an uprojection to the original dimensionality of layer h. Also, a residual connection is added. By the down- and upsizing (instead of inserting a matrix of dimension $M \in \mathbb{R}^{h \times h}$) it is ensured that only a few weights are added.

Invertible adapters are constructed similarly as language adapters, but they aim to capture "token-level language-specific transformations" (Pfeiffer et al., 2020b). They are inserted on top of the embedding layer. However, since the input and output embedding layers are tied, the inverse of the invertible adapter that is stacked on top of the input embedding layer is stacked on top of the output embedding layer. Figure 4 shows where the different adapters are added in the transformer architecture.

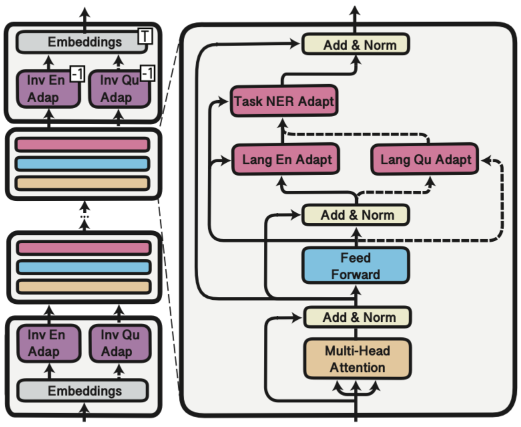


Figure 4: MAD-X architecture as shown in Pfeiffer et al. (2020b). The invertible adapters are placed on top of the input embedding layer and before the output embedding layer (see left side of figure). The task and language adapters are placed inside the transformer layers (one of these layers is exemplified here).

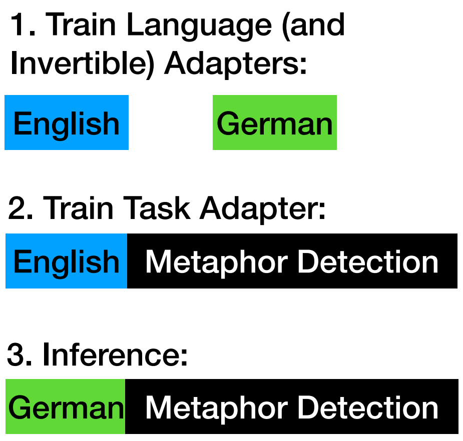


Figure 5: Step by step illustration for using the language and task adapters of MAD-X.

Figure 5 illustrates how the different adapters are used (in the example, only German is depicted as a target language, but the same method is used for every target language). Firstly, the language adapter A_{lang} and the invertible adapter A_{inv} are trained on source (here: English) and target language data (here: Geman) for the

MLM task. Secondly, a task adapter for MD A_{MD} is trained on an English language metaphor dataset (while using the English language and invertible adapters). Thirdly, for zero-shot inference, the English language adapter and invertible adapters are replaced by German language and invertible adapters, while the (language agnostic) task adapter A_{MD} is kept in place.
As the pretrained model that is frozen during the adapter training we use mBERT, and the task adapter is trained by adapting the procedure from Ma et al. (2021) as explained in Section 4.2. We use the the MAD-X implementation found at the AdapterHub library (Pfeiffer et al., 2020a). Here, language adapters are readily available for our target languages German, Russian and Latin. They have been trained on Wikipedia for the given languages. The task adapter for MD was not available and therefore had to be trained on the English training dataset (see Section 3.1.1). For that, the code from AdapterHub had to be adapted for metaphor detection, a step by step direction for adapting the original code for new tasks is to be found online[27].

5 Experiments and Results

The models described in Section 4 are used as the basis for our own experiments. This section firstly covers the basic experimental setup that was used to answer our research questions and its results[28] (see Section 5.1). The following sections show how the results of the basic experimental setup change

- when hyperparameters are tuned (Section 5.2),
- when a larger dataset is used to train the models (Section 5.3),
- when different shots are used for few-shot classification with mBERT (see Section 5.4), and

[27]`https://docs.adapterhub.ml/training.html`
[28]On the reproducibility of the results see Appendix A.3.

- when the datasets are more comparable in terms of linguistic structure (see Section 5.5).

While the first three steps aim at improving model performance, the last step is carried out in order to obtain a clearer picture of the impact of language typology and the amount of pretraining data on model performance. As a last step, we make a brief digression to evaluate the hyperparameter-tuning that was performed (see Section 5.6). All results presented in this section are discussed in Section 6.

5.1 Basic Experimental Setup

Our first research question is the following: *Neural cross-lingual transfer methods have been shown to perform well on a wide range of tasks (such as named entity recognition and question answering). Are they also applicable to MD?* In order to answer this question, we used the target language datasets for Russian (see Section 3.2), German (see Section 3.3) and Latin (see Section 3.4). As the source language dataset we used the English dataset presented in Section 3.1.1. We refer to this dataset by Tsvetkov et al. (2014) as the basic training set. Then we explored how each of the following neural cross-lingual classifiers performed on each of the target languages:

- zero-shot mBERT (see Section 4.2),
- few-shot mBERT with a second fine-tuning on 20 instances[29] of the target language data (see Section 4.2), and
- MAD-X (see Section 4.3).

As hyperparameters for zero- and few-shot mBERT in the basic experimental setup we used the default hyperparameters provided by Huggingface[30], namely (among

[29]The 20 instances are taken from the test datasets, so for mBERT20 the test datasets are slightly reduced compared to the test datasets used for the other experiments.

[30]`https://huggingface.co/docs/transformers/v4.28.1/en/main_classes/trainer#transformers.TrainingArguments`

others) a batch-size of 8, a learning rate of 5e-5, 3 training epochs and the AdamW optimizer. As hyperparameters for MAD-X we used the hyperparameters that Pfeiffer et al. (2020b) mention in their paper, namely a learning rate of 1e-4, a batch-size of 8 and 100 training epochs. In order to make the results reproducible, the seed was set to 42.

The second research question is: *There are languages where only a small amount of pretraining data is available for large language models and/or that are typologically distant from the source language. Can a non-neural classifier outperform neural models for these languages?* In order to answer this question, the random forest classifier described in Section 4.1 was not only used to classify our dataset for Latin, but also the datasets for Russian and German for comparison. As hyperparameters we used the default hyperparameters from *scikitlearn*, namely (among others) 100 estimators, i.e. 100 decision trees, no max-depth limit and Gini as split criterion. In order to make the results reproducible, the seed was set to 1234.

The third and fourth research questions aim at investigating whether the performance of the random forest classifier can be improved by means of feature engineering: *It has been shown that sentences containing metaphorical language seem to be more emotionally charged than non-figurative sentences (Mohammad et al., 2016). Will adding emotion scores as conceptual features improve the performance of the non-neural classifier?* and *Research has shown that mid-range abstractness ratings are unreliable, since they exhibit a large degree of disagreement among annotators (Pollock, 2018). Will separating mid-range abstractness ratings from truly concrete and abstract ratings improve the performance of the non-neural classifier?* Therefore, in addition to reproducing the results from Tsvetkov et al. (2014) (see Section 4.1), we firstly tested whether or not adding emotional scores (valence, dominance and arousal) collected by Mohammad (2018) as real-numbered features improves the performance of the original classifier. Secondly, it was tried out whether replacing binary abstractness scores by three categories ("abstract", "neutral" and "concrete") would improve the original classifier's performance, since in this setting unreliable mid-range ratings are treated differently from truly abstract and concrete ratings.

The classifier, which was reproduced as closely as possible to that of Tsvetkov et al.

(2014), contained 281 features altogether (see Table 3). For the newly added feature type emotion (VAD), 9 features were added, namely one feature for valence, arousal and dominance for each of the three syntactic functions subject, verb and object ($3 \times 3 = 9$). An overview over the numbers of features for the original random forest classifier with added VAD-scores is given in Table 4.

feature type	# **features**
Abstractness/Imageability	22
Supersenses	67
VSM	192
VAD	9
All	290

Table 4: Table presenting the numbers of features used for the random forest classifier for each feature type when the model by Tsvetkov et al. (2014) is changed by adding emotion scores (VAD).

Replacing binary abstractness scores by 3 categories led to a total number of 300 features, where 41 features were used for abstractness and imageability instead of the original 22 features in the classifier from Tsvetkov et al. (2014).
In theory, the following 54 features are possible for abstractness and imageability in the scenario with three labels (*a* stands for abstract, *c* for concrete, *m* for midrange values) rather than 41:

- 9 individual features for abstractness:
  subject_a, verb_a, object_a, subject_m, verb_m, object_m, subject_c, verb_c, object_c
- 9 individual features for imageability: the same categories as for abstractness
- 18 combinations of features for abstractness:
  subject_verb_aa, subject_verb_am, subject_verb_ac, subject_verb_ma, subject_verb_mm, subject_verb_mc, subject_verb_ca, subject_verb_cm,

subject_verb_cc, verb_object_aa, verb_object_am, verb_object_ac, verb_object_ma, verb_object_mm, verb_object_mc, verb_object_ca, verb_object_cm, verb_object_cc

- 18 combinations of features for imageability: the same features as for abstractness

However, when mid-range ratings were introduced, some combinations from above (for example subject_verb_cc for imageability) and some of the new combinations (for example verb_object_cm for imageability) no longer occurred in the training data and were therefore not used as features for inference[31]. Only 41 of the features occurred in the training data, as shown in Table 5, which lists the feature types for the original random forest classifier with three instead of two abstractness scores. In cases, where the difference between the results for two models was very small and where it was deemed important for the interpretation of the results, significance testing with χ^2 was carried out. This holds for the current and the following subsections.

feature type	**# features**
Abstractness/Imageability_3	41
Supersenses	67
VSM	192
All	300

Table 5: Table presenting the numbers of features used for the random forest classifier for each feature type when the model by Tsvetkov et al. (2014) was changed by using three categories (abstract, medium, concrete) instead of two (abstract, concrete).

The results of the basic experimental setup (default hyperparameters and basic training data) can be found in Table 6. It shows the F1-scores for detecting non-literal

[31]The feature number 300 assumes thresholds of 0.7 and 0.3 for the abstractness and imageability scores, i.e. abstract: rating higher than 0.7, medium: rating between 0.3 and 0.7, and concrete: rating below 0.3).

	Russian	**German**	**Latin**
majority vote	66.7	66.7	66.7
mBERT0	**86.7**	**77.6**	69.1*
mBERT20	82.9	66.6	30.8
MAD-X	56.8	52.0	18.2
random forest	80.8	70,8	66.7
+ **VAD**	78.7	71.7	**70.6****
+ **abstr_imag_3**	80.4	70.3	68.4

Table 6: F-scores for detecting non-literal verb usages using the basic training dataset by Tsvetkov et al. (2014) and default hyperparameters for all models. Model with asterisk (*) performs significantly better than the baseline; model with double asterisk (**) performs significantly better than model with one asterisk (*) according to χ^2 testing ($p<0.05$).

word usage of the target word for the four models: zero- and few-shot classification with mBERT (mBERT0 and mBERT20, respectively), zero-shot classification with MAD-X and classification with the random forest classifier (once in the vanilla version that closely resembles the model from Tsvetkov et al. (2014), once boosted with VAD-scores and once with three abstractness and imageability labels instead of two (abstr_img_3)). The performance of each model is shown for each of the three target languages, Russian, German and Latin. The baseline is majority vote[32]; for the balanced datasets, this leads to an F1-score of 66.7. This layout is the basis for all results presented in this section, so this description will not be repeated. The best-performing model for Russian (86.7) and German (77.6) in this setup is zero-shot classification with mBERT. For Latin, the random forest classifier boosted with VAD-scores performs best (70.6). It is closely followed by mBERT0 (69.1), which according to χ^2 testing performs significantly better than the baseline, while the random forest classifier with VAD performs significantly better than mBERT0 ($p<0.05$).

[32]In the case of a balanced dataset, majority vote is used in the sense that all predictions are "metaphorical".

Adding a second round of fine-tuning (few-shot setup) decreases the F1-scores across all languages; for German and Latin, the performance is below the baseline. MAD-X performs considerably worse than the baseline across all languages.
The vanilla random forest classifier performs better than the baseline for Russian and German, but on par with the baseline for Latin. Adding VAD-scores improves the performance of the random forest classifier by 0.9 points for German and 3.9 points for Latin. For Russian, adding VAD-scores leads to worse results than using the vanilla random forest classifier. When using three abstractness labels instead of two, the performance slightly deteriorates compared to the vanilla classifier for Russian and German, while the F1-score for Latin increases by 1.7 points.

5.2 Performing Hyperparameter-Tuning

When dealing with high-resource languages, the data is usually split into train, dev (or validation) and test set. For training (or fine-tuning) the model, the train set is used. For fine-tuning the hyperparameters, the dev set is used: Different sets of hyperparameters are tried out on the dev set, and the optimal hyperparameters found for the dev set are used for evaluating the performance of this model on unseen test data, namely the test set. This is done in order to confirm that the hyperparameters found for the dev set generalize to unseen data and that no overfitting to the particular dev set is occuring.
In this thesis, classifiers for low-resource languages, where there is no or very little training data available and definitely no validation data, are explored. Therefore, the following procedure was tried out: the models were trained on the basic English dataset for different sets of hyperparameters. Then these different hyperparameter sets were evaluated on a different MD dataset from the high resource source language English, namely the dataset by Mohammad et al. (2016). For a description of this dataset see Appendix A.7. This English dataset was used as the dev set, since no dev set from the target language was available. Finally, the hyperparameter set that led to the best performance on the dev set (i.e. the hightest F1-score for detecting non-literal word usage) was used for testing the model on unseen test data, namely

the datasets for the three languages Russian, German and Latin.
The hyperparameter search was carried out in the form of a grid search, where different combinations of hyperparameters were tried out. Table 7 reports the hyperparameters[33] that were used for this grid search, while Table 8 reports the hyperparameters that led to the hightest F1-score for detecting non-literal verb usages during the grid search on the English dataset by Mohammad et al. (2016).

	learning rates	**epochs**	**batch size**
mBERT0 and mBERT20	1e-4, 1e-5, 1e-6	8, 16, 32	8, 16, 32
MAD-X	1e-3, 1e-4, 1e-5	10, 50, 100	8, 16, 32
	# of estimators	**max tree depth**	**seed**
random forest	10, 50, 100	None, 5, 10	83, 297, 1234

Table 7: Hyperparameter values used for the grid searches for the different classification models.

	learning rate	**epochs**	**batch size**
zero	1e-6	8	32
few_ru	1e-6	8	16
few_ge	1e-6	8	32
few_la	1e-6	8	32
madx	1e-3	50	8
	# of estimators	**max tree depth**	**seed**
random forest	10	None	297

Table 8: Hyperparameter sets that were found to lead to the highest F1-score for detecting non-literal word usage during the grid search.

To sum up: The setup is identical to the basic experimental setup described in Section 5.1; however, the hyperparameter sets that led to the best performance during

[33]See Appendix A.9 for an explanation of why the hyperparameter values given in Table 7 were chosen for the grid search.

the grid search were used instead of the default hyperparameters.
The following paragraph presents the results of the hyperparameter-tuning: The hyperparameters shown in Table 8 form the basis for the results presented in Table 9. Here, the basic training dataset was used as well as the best-performing hyperparameter sets obtained from grid search. The best results for Russian and German are achieved by MAD-X, while for Latin the random forest classifier boosted with VAD-scores performs best. Zero- and few-shot classification with mBERT performs worse or just slighty better than the baseline across all three languages. MAD-X performs well for Russian and German, while performing slightly better than the baseline for Latin. The vanilla random forest classifier performs slightly better than the baseline for Russian and German, while performing worse than the baseline for Latin. Adding VAD-scores improves the F1-scores across all languages here. Using three abstractness labels instead of two improves the performance compared to the vanilla random forest classifier for Russian and Latin, but for German performance slightly worsens. Compared to the results for the default hyperparameters (see Table 6), the performance of all models decreases, but for MAD-X the performance improves drastically.

	Russian	**German**	**Latin**
majority vote	66.7	66.7	66.7
mBERT0	66.7	63.9	58.8
mBERT20	69.6	64.6	63.8
MAD-X	**81.9**	**73.7**	68.0
random forest	73.3	69.0	61.8
+ **VAD**	78.3	70.3	**68.9**
+ **abstr_imag_3**	76.2	68.1	67.8

Table 9: F-scores for detecting non-literal verb usages with the best results obtained from hyperparameter-tuning. As training data, the basic training dataset by Tsvetkov et al. (2014) was used.

5.3 Augmenting the Amount of Training Data

In order to investigate whether augmenting the amount of training data improves the results, the dataset by Tsvetkov et al. (2014), which consists of 222 instances, was augmented by the dataset by Mohammad et al. (2016), which consists of 1639 instances, so that the number of training instances was 1861 instead of 222. We will refer to this combined dataset of 1861 instances as the augmented dataset, as opposed to the basic training set which includes only the data by Tsvetkov et al. (2014). The setup is identical to the basic experimental setup described in Section 5.1; however, instead of the basic dataset by Tsvetkov et al. (2014), the augmented dataset is used.

	Russian	**German**	**Latin**
majority vote	66.7	66.7	66.7
mBERT0	**90.5**	**76.8***	63.8
mBERT20	87.6	76.6	31.4
MAD-X	86.2	75.1	59.2
random forest	85.5	71.0	**72.2**
+ **VAD**	85.8	71.5	66.7
+ **abstr_imag_3**	85.1	70.3	67.2

Table 10: F-scores for detecting non-literal verb usage using the <u>default hyperparameters</u> for all models and the <u>augmented training dataset</u> (Tsvetkov et al. (2014) and Mohammad et al. (2016)). The model with asterisk (*) performs significantly worse than the corresponding model with the basic training dataset.

The results are shown in Table 10. Zero-classification with mBERT performs best for Russian (90.5) and for German (76.8). However, the performance for German is – according to χ^2 ($p<0.05$) – significantly worse than the corresponding experiment with the basic training dataset. The best performance for Latin is achieved by the vanilla random forest classifier (72.2). While mBERT0 performs well for German and Russian, the results for Latin are below the baseline. As before, the performance of mBERT drops when a second round of fine-tuning is added: mBERT20 performs worse in all three languages than mBERT0, but still slightly better than MAD-X (except for Latin, where all neural models perform below the baseline). The vanilla random forest classifier performs worse than the neural models for Russian and German, while better than the neural models for Latin. Adding VAD-scores improves the performance of the classifier slightly for Russian and German (< 1 point), but decreases the F1-score for Latin by almost 5 points. As before, using three abstractness and imageability labels leads to a decline in the F1-score compared to using two labels as in the vanilla classifier.

5.4 Selecting Shots in Few-Shot Classification

For the few-shot classification with mBERT in the basic experimental setup, a random split of the test data was used for the second fine-tuning (see Table 6). The results were not satisfactory in two ways: Firstly, in contrast to what was expected, the second-fine-tuning led to a decrease in the F1-score compared to the zero-shot classification with mBERT. For example the F1-score for Russian dropped from 86.7 to 82.9, even though the training conditions improved quantitatively (there were 20 more training instances) and qualitatively (the 20 additional training instances came from the target language). Secondly, the results for German and Latin were very low, as both did not reach the majority vote baseline. Augmenting the training data led to better results for German and Russian compared to using the basic training dataset, but the F1-score for Latin was still very low and all few-shot results were still lower than the zero-shot results.
To investigate whether or not some shots lead to better performance than others, i.e. whether it is important to actively select the shot that leads to good performance, few-shot classification was carried out 5 times, each with a different split of the target language dataset. For each model, the mean F1-score across the different shots was calculated as well as the standard deviation in order to see how much influence the selection of the shot has on model performance.
This experiment was carried out once with the basic training set from Tsvetkov

et al. (2014) and once with the augmented dataset (Tsvetkov et al. (2014) and Mohammad et al. (2016)). The experimental setup is therefore identical to the few-shot setups described in Sections 5.1 and 5.3, but instead of using one random shot, 5 different shots were used. The results (F1) for the individual shots as well as the mean F1-score and the standard deviation for each language are given in Tables 11 and 12. Table 11 reports the numbers for using the basic training dataset, while Table 12 reports the numbers for the augmented dataset.

	indiv. F1	**mean**	**stand. dev.**
	87.3		
	50.3		
Russian	80.0	76.3	15.1
	77.8		
	86.1		
	65.3		
	80.9		
German	77.2	75.2	6.0
	78.0		
	74.4		
	0		
	66.7		
Latin	62.3	51.8	29.0
	63.2		
	66.7		

Table 11: Individual F1-scores, mean and standard deviation for using 5 different shots of the target language datasets for the second fine-tuning. The first fine-tuning was carried out with the basic English dataset from Tsvetkov et al. (2014).

For the **basic English dataset** as training set, the mean F1-score across the five shots for Russian is 76.3 and is therefore slightly higher than the mean F1-score for German (75.2). For Latin, the mean F1-score is 51.8. The standard deviation is very high for Latin (29.0), which is, however, due to only one outlier (0). The other F1-scores range between 62.3 and 66.7. The standard deviation is rather high for Russian (15.1), and slightly lower for German (6.0). The maximum value for Russian in the individual F1-scores is 87.3, which is higher than the F1-score for zero-shot classification with the basic training set and default hyperparameters, while the lowest F1-score is 50.3, which is clearly below the baseline. For German, the best shot led to a performance of 80.9, which is the highest F1-score that was achieved for this dataset. The lowest result for German is also below the baseline (65.3). For Latin, all results are close to the baseline or below.

	indiv. F1	**mean**	**stand. dev.**
Russian	86.2 81.6 85.6 84.2 82.9	84.1	1.9
German	68.6 80.7 79.9 75.2 78.3	76.5	4.9
Latin	57.5 65.9 64.0 66.7 60.8	63.0	3.8

Table 12: Individual F1-scores, mean and standard deviation for using 5 different shots of the target language datasets for the second fine-tuning. The first fine-tuning was carried out with the augmented English dataset.

For the **augmented English dataset** as training set, the mean F1-score across all languages is slightly higher than for the basic training set, while the standard deviation is substantially lower. The maximum value among the individual F1-scores is 86.2 for Russian and 80.7 for German, so both values are slightly lower than the maximum values for the basic training dataset. For Latin, the scores still do not exceed the baseline.

5.5 Making Training Data More Comparable

When comparing the dataset for the target languages, it becomes obvious that in many German and Latin sentences there is only a verb (i.e. a target word), but no subject or object dependent on the target word, whereas in the Russian dataset for each target word there is at least a subject or an object (see also data description in Section 3). Especially for the random forest classifier, which only uses the lemmatized subjects, verbs and objects as a basis for feature generation, this is a major problem: The verb itself does not carry enough information for the classifier to make an informed decision. In order to see how the different classifiers perform on more comparable datasets, the German and Latin datasets were reduced by removing sentences

- where the target word (i.e. the verb) comes without a subject and without an object,
- where besides the verb only an adjective (such as *omnis* – "every(one)") or a pronoun is given as subject or object, and
- where information about the metaphoricity of the target word does not lie within the subject or the object but another part of the sentence. Here are two examples:

  (25) accipietis donum Spiritus Sancti (translation: "you will receive the gift of the Holy Spirit")

  (26) die Sprosse der Karriereleiter ansägen (translation: "to saw the rung of the career ladder")

In both sentences, the object is not enough to make an informed classification decision, but the attribute of the object (*of the Holy Spirit* and *of the career ladder*) is necessary for determining whether the verb is used literally or not.

The reduced dataset for German consists of 228 sentences, of which 110 are labelled as non-literal and 118 are labelled as literal, while the reduced Latin dataset consists of 50 sentences, of which 27 are labelled as non-literal and 23 are labelled as literal[34]. Then we used default hyperparameters and the augmented dataset, so the experimental setup described in Section 5.3, on these reduced dataset versions for German and Latin (as Russian is the dataset that we are trying to emulate, this dataset remains unchanged).

	Russian	**German**	**Latin**
majority vote	66.7	0	0.7
mBERT0	90.5	**82.5**	60.0
mBERT20	87.6	74.9	72.2
MAD-X	86.2	78.8	63.8
random forest	85.5	78.0	**73.3**
+ **VAD**	85.8	78.5	62.1
+ **abstr_imag_3**	85.1	76.5	64.5

Table 13: F1-scores for non-literal verb usages using the default hyperparameters for all models and the augmented training dataset. The datasets for German and Latin are reduced in such a way that the linguistic structure is more comparable to the Russian dataset. As the results for Russian were not repeated but taken from Table 10 for comparison, they are given in gray.

The results are shown in Table 13. The best-performing model is mBERT0 for German (F1: 82.5) and the vanilla random forest classifier for Latin (F1: 73.3). mBERT0 does not perform well for Latin, as its F1-score is below the baseline. For Latin,

[34]Due to this imbalance the baseline for the reduced datasets for German and Latin is no longer 66.7, but 0 and 0.7, respectively.

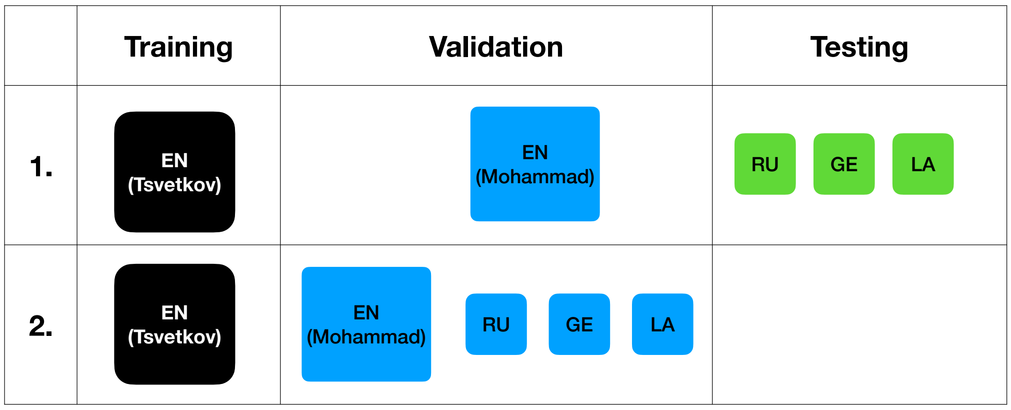
mBERT20 achieves an F1 of 72.2, which is higher than the result for zero-shot classification with BERT, but for German the additional fine-tuning leads to a worse result than the zero-shot classification. MAD-X performs better than mBERT20 for German (but worse than BERT0), while for Latin MAD-X performs below the baseline.
The vanilla random forest classifier performs slightly worse than MAD-X for German, while for Latin it leads to an improved result. Adding VAD-scores improves the F1-score for German, but the performance deteriorates for Latin. Using three abstractness and imageability labels leads to a worse performance than using two labels in both languages.
When comparing the numbers for the reduced datasets with Russian, i.e. when comparing the datasets that are more comparable than they were in the previous experiments, it is obvious that Russian still performs best among the three languages, while German takes the second place and Latin the third place.
In comparison to previous experiments, German and Latin achieve the highest score of all experiments.

5.6 Digression: Evaluating Hyperparameter-Tuning

5.6.1 Using Source Language Validation Data

By fine-tuning the pretrained language model, the model not only learns to perform the task itself, but is also fine-tuned to work well for a given language and a given domain. Therefore, it is not clear whether finding the optimal hyperparameters for a dataset in the source language also leads to the optimal hyperparameters for a dataset in the target language. In order to see whether the procedure that we employed in Section 5.2 is valid, we evaluated it by comparing the hyperparameter sets that are found for English with the hyperparameter sets that would have been found for German, Russian and Latin – if the datasets that we so far used as evaluation data were used as <u>validation data</u>. The course of action is illustrated in Figure 6 and explained in a more detailled way in the following.
As a **first step** (see first row in Figure 6), we trained each classifier on the basic

English training dataset for each hyperparameter set, chose the hyperparameter set with the best performance in terms of F1 score for detecting non-literal verb usage on the dev set, and evaluated this model on the test sets for the target languages. This procedure was described in Section 5.2.



	Training	Validation	Testing
1.	EN (Tsvetkov)	EN (Mohammad)	RU GE LA
2.	EN (Tsvetkov)	EN (Mohammad) RU GE LA	

Figure 6: Course of action for evaluating whether a source language dataset can be used to fine-tune hyperparameters in a cross-lingual setup. The first line illustrates the procedure employed in Section 5.2, while the second line illustrates how this procedure is evaluated in this section.

In the **second step** (see second row in Figure 6) – for evaluating the procedure shown in the first row, which is what we do in this section –, we obtained the results of the grid search for each model with the English dev set and with the target language datasets. Finally, we examined whether the F1-scores for the different hyperparameter sets correlate for the English dev set and each of the target language datasets[35]. A strong correlation between the performance of the source and target

[35]As we picked the best hyperparameter set for the dev set already in step one (see Section 5.2)

language datasets would be an indicator that fine-tuning the hyperparameters on the source language is enough and no target language material is necessary for the validation process.

For zero- and few-shot mBERT, the evaluation of the grid search was conducted as described so far and as can be seen in the second row of Figure 6. For MAD-X, the procedure shown in the second row was carried out for the task adapter only, since the language and invertible adapters were used off-the-shelf from AdapterHub. For the random forest classifier as shown in Section 2.2, data was needed where subject, verb and object were marked and lemmatized. Since obtaining this information for the whole dataset by Mohammad et al. (2016) would have been very time-consuming, only a small subset of this dataset, namely 100 samples, was used as dev set, where lemmatized subject, verb and object were annotated manually.

– without having seen the performance of the test sets for the different hyperparameter sets –, the results that are shown in Table 9 are obtained from truly unseen data.

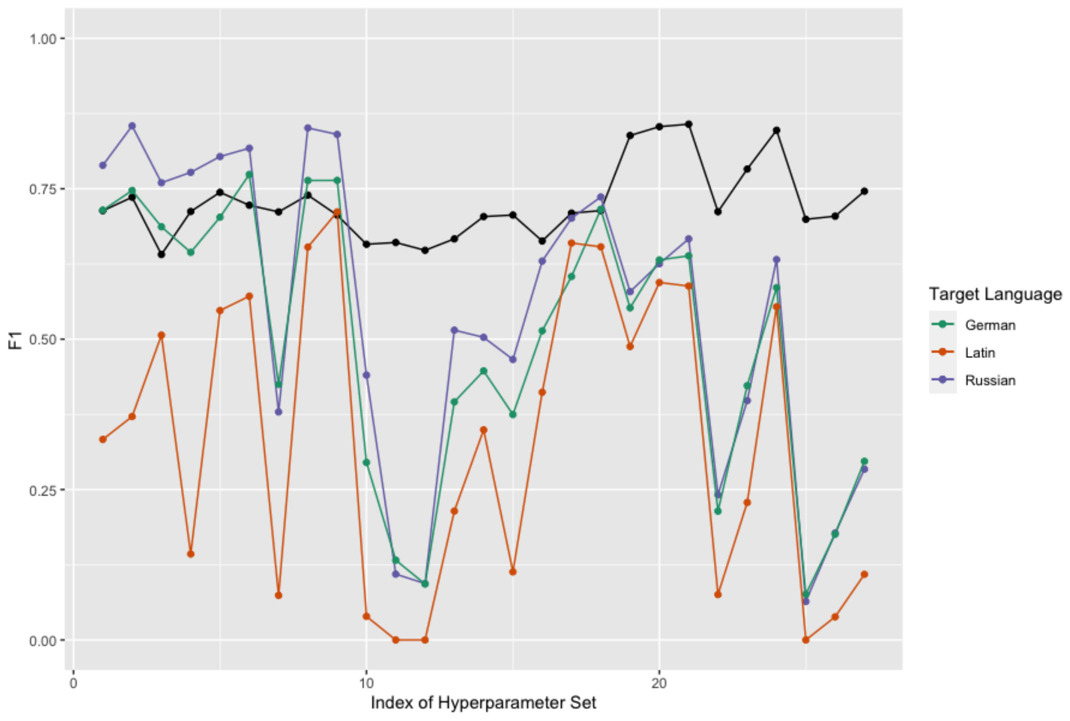


Figure 7: Result for using both the data from Mohammad et al. (2016) (black line) and the different test sets for target languages Russian, German and Latin as dev sets for the grid search on zero-shot classification with mBERT.

In the following, the results for the experiments illustrated in the second row of Figure 6 are given. For each classification model, it is depicted how different sets of hyperparameters influence model performance when using validation data from English compared to using (simulated) validation data from the target languages. The results for evaluating the hyperparameter-tuning for **zero-shot classification with mBERT** are illustrated in Figure 7. It shows the F1-scores for detecting non-literal word usage (y-axis) for the different hyperparameter sets that were tried out during the grid search, where each index on the x-axis corresonds to a particular hyperparameter set. For the mapping between index and hyperparameter set see Tables 19 (zero- and few-shot mBERT), 20 (MAD-X) and 21 (random forest) in Appendix A.6. Figure 7 compares the performance of each hyperparameter set on the English validation set (black line) with the performance of the hyperparameter

sets for (simulated) validation sets in the target languages German, Latin and Russian (lines in green, orange and blue). The results presented in Table 7 are discrete; the lines have been added for reasons of perspicuity.
For the English validation dataset, the maximum F1-score is reached with hyperparameter set 21 (learning rate: 1e-6, epochs: 8, batch size: 32) and the top 3 results for the English validation dataset are reached with a learning rate of 1e-6[36]. For Russian, the maximum F1-score is reached with hyperparameter set 2 (learning rate: 1e-4, epochs: 8, batch size: 16), for German with hyperparameter set 6 (learning rate: 1e-4, epochs: 16, batch size: 32) and for Latin with hyperparameter set 9 (learning rate: 1e-4, epochs: 32. batch size: 32). For Russian and German, the top 3 results are achieved by a learning rate of 1e-4, while for Latin the top 3 results are achieved with a learning rate of 1e-4 or 1e-5.
The results for evaluating the hyperparameter-tuning for **few-shot classification with mBERT** are illustrated in Figure 8. Whereas for zero-shot classification generally one model is trained and used for inference on all languages, in few-shot classification each language has its own model. This is why the results are compared for all languages within one figure for zero-shot classification and in three different subfigures for few-shot classification. In theory, one could also perform hyperparameter-tuning for the German few-shot classifier on the Russian and Latin datasets. However, as the primary aim is to evaluate whether performing hyperparameter-tuning on the high-resource source language is valid, only the comparison between the target language that the few-shot classifier was trained on and English is made.

[36]Indices 1 - 9 represent hyperparameter sets with a learning rate of 1e-4, indices 10-18 represent sets with a learning rate of 1e-5 and indices 19-27 represent sets with a learning rate of 1e-6.

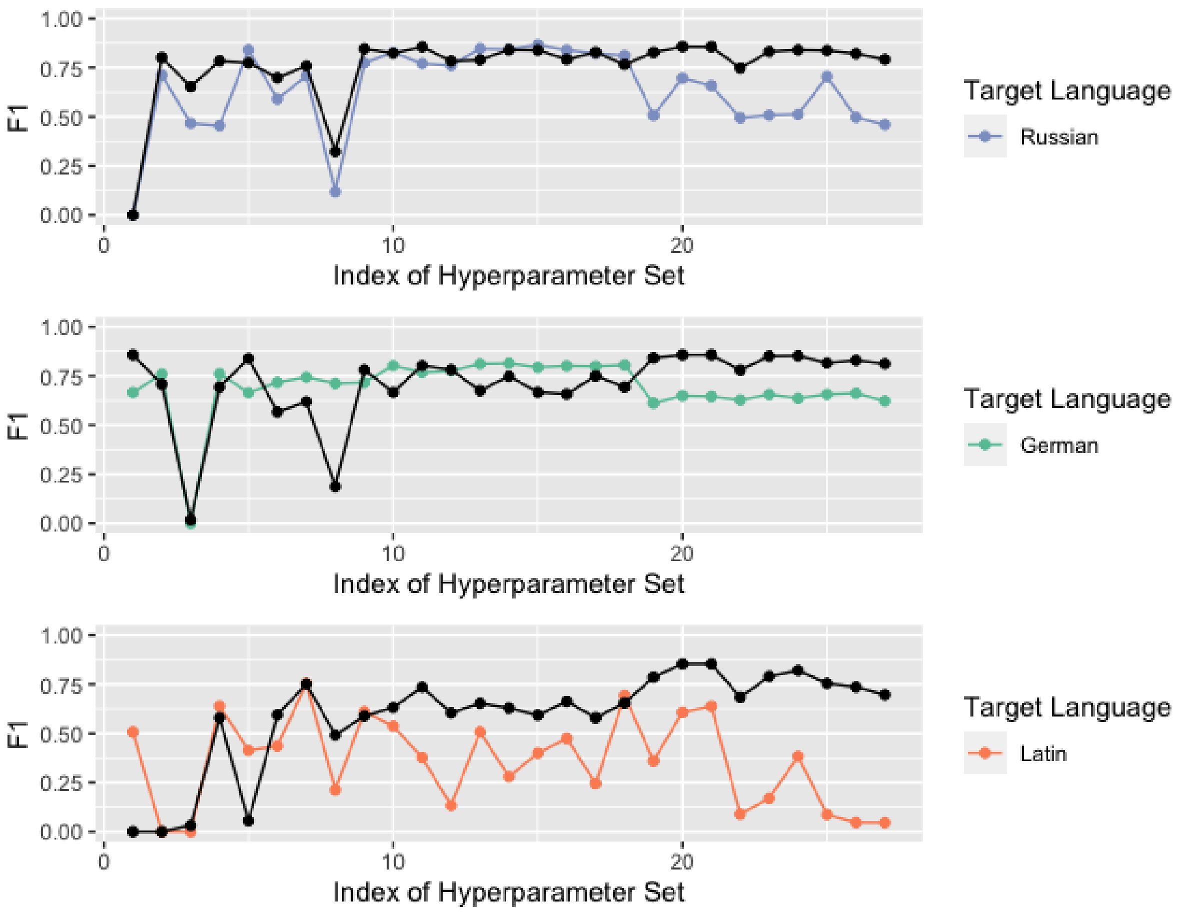


Figure 8: Result for using both the data from Mohammad et al. (2016) (black line) and the different test sets for target languages Russian, German and Latin as dev sets for the grid search on few-shot classification with mBERT.

For the Russian few-shot model, the maximum F1-score is reached by hyperparameter set 20 (learning rate: 1e-6, epochs: 8, batch-size: 16) for English and by hyperparameter set 15 for Russian (learning rate: 1e-5, epochs: 16, batch-size: 32). For a learning rate of 1e-5 (index 10-18) both models show a stable performance clearly above the baseline.

For the German few-shot model, the maximum F1-score is reached by hyperparameter set 14 (learning rate: 1e-5, epochs 16:, batch size: 16) for German. For English it is achived by hyperparameter set 20 (learning rate: 1e-6, epochs: 8, batch size:

16). With one exception, all hyperparameter sets lead to a performance above the baseline for German, especially a learning rate of 1e-5, which leads to a stable performance at a high level. For English, the best and most stable performance is achieved by a learning rate of 1e-6.

For the Latin few-shot model, the maximum F1-score is reached by hyperparameter set 7 (learning rate: 1e-4, epochs: 32, batch size: 8) for Latin and by hyperparameter set 21 (learning rate: 1e-6, epochs: 8, batch size: 32) for English. The results for Latin are very unstable, only two hyperparameter sets lead to a performance above the baseline. While these two results are achieved by a learning rate of 1e-4 and 1e-5 respectively, the top 3 results for English are achieved by a learning rate of 1e-6.

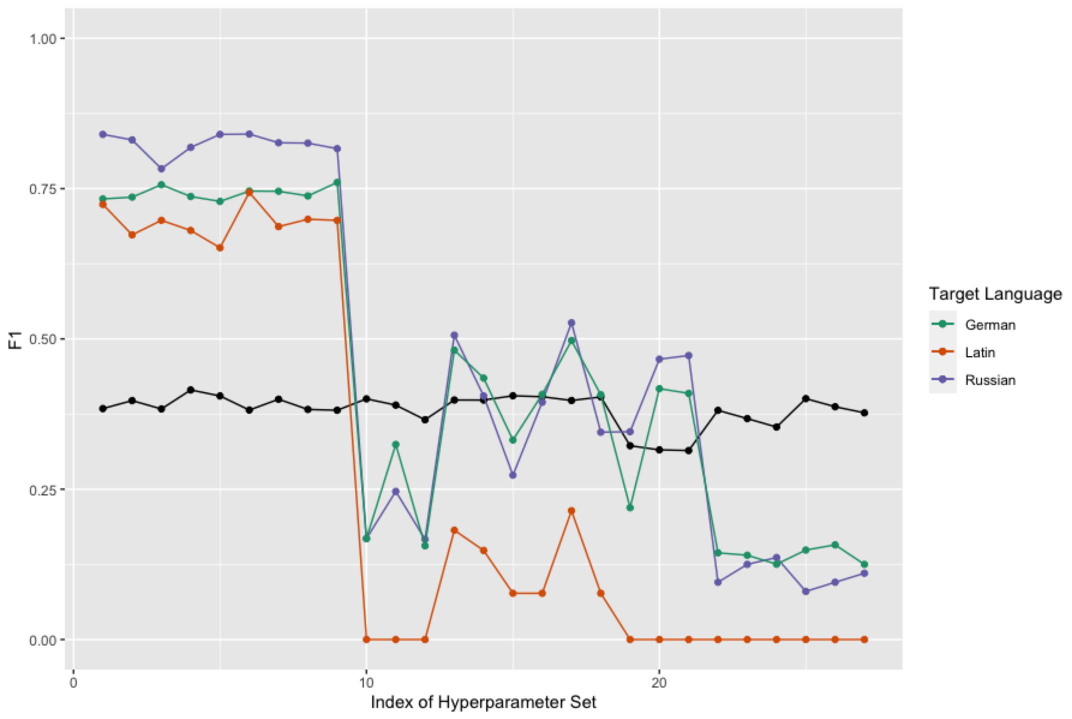


Figure 9: Result for using both the data from Mohammad et al. (2016) (black line) and the different test sets for target languages Russian, German and Latin as dev sets for the grid search on zero-shot classification with MAD-X.

The results for evaluating hyperparameter-tuning for **MAD-X** are illustrated in Figure 9. For the English validation set, the top F1-score is reached by hyperparameter set 4 (learning rate: 1e-3, epochs: 50, batch size: 8), for Russian with hyperparameter set 6 (learning rate: 1e-3, epochs: 50, batch size: 32), for German with hyperparameter set 9 (learning rate: 1e-3, epochs: 50, batch size: 8) and for Latin with hyperparameter set 6 (learning rate: 1e-3, epochs: 50, batch size: 8). The results for English are stable at a low level[37], while for the target languages stable above baseline performance is reached by a learning rate of 1e-3 (one exception is hyperparameter set 5 which leads to lower than baseline performance for Latin). The two other learning rates that were tried out (1e-4 and 1e-5) led to performances below the baseline for the target languages.

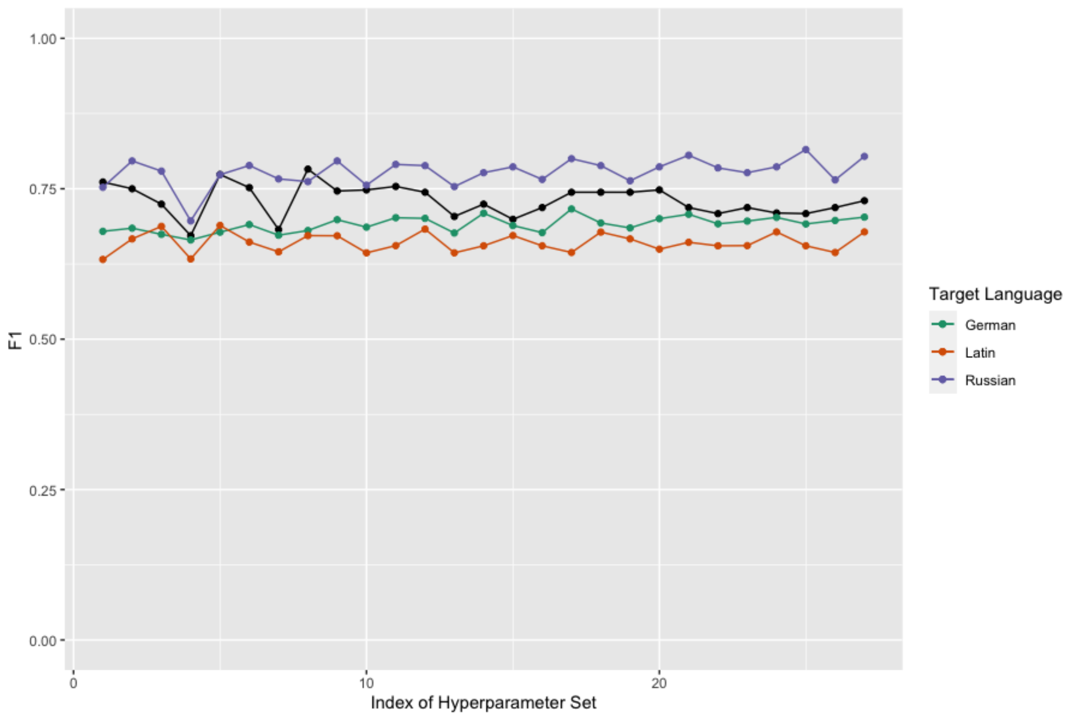


Figure 10: Result for using both the data from Mohammad et al. (2016) (black line) and the different test sets for target languages Russian, German and Latin as dev sets for the grid search on zero-shot classification with the random forest classifier.

[37]It should be noted here that due to the imbalance of the dataset by Mohammad et al. (2016), the majority vote baseline for this validation set is 0.

The results for evaluating the hyperparameter-tuning for the **random forest classifier** are shown in Figure 10. The maximum F1-score for English is achieved by hyperparameter set 8 (number of estimators: 10, max depth: None, random state: 297), for Russian by hyperparameter set 25 (number of estimators: 100, max depth: None, random state: 83), for German by hyperparameter set 14 (number of estimators: 50, max depth: 10, random state: 297) and for Latin by hyperparameter set 3 (number of estimators: 10, max depth: 5, random state: 1234). The performance of the models is very stable across hyperparameter sets. Except for Latin, all languages outperform the baseline. For Latin the models perform closely around baseline performance, with some results being slightly lower and some slightly higher than the baseline.

	Russian	**German**	**Latin**
mBERT0	0.35	0.43	0.49
mBERT20	0.77	0.51	0.24
MAD-X	0.18	0.24	0.34
RF	0.08	0.12	0.24

Table 14: Spearman's rank order correlation between the F1-scores obtained during the grid search for the English validation set and the three target language datasets.

In Figures 7, 8, 9, and 10, the hyperparameter sets for English and the target languages seem to correlate more or less, depending on the language and the model. In order to quantify these first impressions, the correlation coefficient was calculated between the source and target language results shown in the figures. We calculated Spearman's rank order correlation[38] between the results for English and each of the target languages for each model. For this, the results of the different hyperparameter sets were firstly ranked according to their F1 score. Secondly, the following formula

[38]We use Spearman's correlation instead of Pearson's correlation because we are interested in the monotonic relationship, i.e. we want to know for example whether the top results for English are also the top results for German, independent of how much the F1 score varies.

was used to calculate the correlation:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}, \tag{3}$$

where d is the difference between each of the two ranks and n is the number of observations. Spearman's ρ indicates the strength and the direction of the association between two variables. A value of 1 (or -1) indicates a perfect positive (or negative) correlation, whereas a value of zero indicates that there is no correlation to be found. The results for calculating Spearman's ρ can be found in Table 14. With the exception of mBERT20 and Russian, all values indicate moderate, low or negligible positive correlation. The lowest correlation is found for the random forest classifier, while the highest correlation is found for zero- and few-shot mBERT.

5.6.2 Using Third Language Validation Data

Originally, we tested whether or not there is a correlation between the results of the hyperparameter-tuning for the English validation set and each of the target language datasets. As a byproduct, Figures 7 and 9 hinted at a strong correlation between the results for the target languages among themselves for zero-shot mBERT and for MAD-X. Therefore, the correlation strength between all languages (in terms of Spearman's rank order correlation) was calculated for these two models. The results are shown in Figures 11 and 12, respectively.

For both zero-shot mBERT and MAD-X, the correlation between English and the target languages is the lowest of all combinations, while the correlation between Russian and German is the highest. The correlation among all the target languages is strongly positive with correlation values ranging from 0.79 to 0.97.

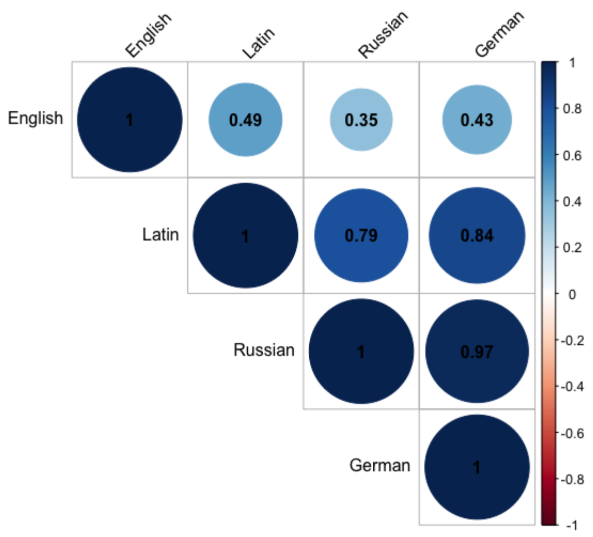


Figure 11: Spearman's rank order correlation between all languages for zero-shot classification with mBERT.

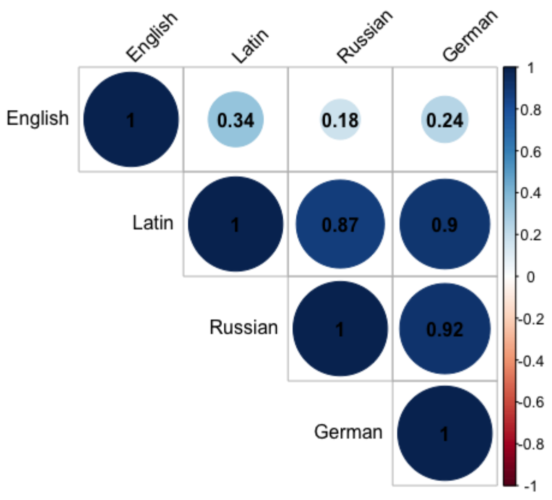


Figure 12: Spearman's rank order correlation between all languages for MAD-X.

6 Discussion

Firstly, the results described in the previous section are discussed here with regard to the research questions of this thesis. Secondly, conclusions that can be drawn from the results that go beyond the research questions are discussed.

6.1 Answering the Research Questions

This section presents the research questions and discusses which answers can be drawn from the results presented in Section 5. In order to do this, we take into account only the results for the basic training set with default hyperparameters (Table 6), the augmented training set with default hyperparameters (Table 10) and the results from Tables 11 and 12, where the influence of shot selection is presented. All other results are discussed in the following Section 6.2.

6.1.1 Research Question 1

Neural cross-lingual transfer methods have been shown to perform well on a wide range of tasks (such as named entity recognition and question answering). Are they also applicable to MD? In order to answer this question, we look at the results that the neural models zero-shot classification with mBERT, few-shot classification with mBERT and MAD-X achieve for the Russian, German and Latin datasets.
Russian: When using the basic training set and default hyperparameters for Russian, the zero-shot mBERT classifier obtains an F1-score that is slightly higher than the result published by Tsvetkov et al. (2014) for their dataset: 86.7 (see Table 6, cf. Tsvetkov et al. (2014): 86.0). Even though adding a second round of fine-tuning with 20 German instances that are randomly sampled reduces the F1-score slightly, using the optimal shot in our experiment gave us an F1-score of 87.3 (see Table 11). Augmenting the basic dataset and using default hyperparameters led to an even higher result of of 90.5 (F1) for the zero-shot mBERT model (see Table 10). This is coming close to the results Ma et al. (2021) published for monolingual MD, where the results reached F1-scores up to 94.45 for English. With MAD-X, a result of 86.2 was reached with the augmented dataset and default hyperparameters (see Table 10), which is also slightly higher than the result published by Tsvetkov et al. (2014). To put it in a nutshell: for Russian, all three neural models outperform the results

published by Tsvetkov et al. (2014), while the zero-shot mBERT classifier that uses the augmented training dataset and default hyperparameters is the most successful model.

German: For German, using the basic training set and default hyperparameters with zero-shot mBERT results in an F1-score of 77.6 (see Table 6). Köper and Schulte im Walde (2016) report an F1-score of 77.3 for detecting non-literal verb usage on their dataset; it has to be mentioned, though, that the results presented here are based on a balanced version of the original dataset by Köper and Schulte im Walde (2016), so the results should not be compared directly. Surprisingly, the performance deteriorates significantly when the training dataset is augmented (according to χ^2, $p<0.05$). Even though the F1-score for mBERT20 is slightly lower than for mBERT0 with a randomly sampled shot, a better shot can lead to an increased performance up to 80.9 (F1, see Table 11). In contrast to Russian, augmenting the training data does not lead to an improvement of the results for German for mBERT0 and mBERT20 compared to using the basic training dataset, while for MAD-X the performance is improved compared to the basic training dataset (75.1 instead of 52.0 (F1), see Table 10). To put it in a nutshell: all models outperform the baseline for German. In general, zero-shot mBERT0 performs best in the standard setup, but few-shot mBERT seems to be a promising candidate for improving the results – if it can be found out which factors constitute a successful shot (which is a topic to be addressed by future work).

Latin: As we just saw, the neural models perform well for Russian and German. For Latin, however, there's a different picture, as expected: Only mBERT0 with default hyperparameters using the basic training set achieves a performance of 69.1 (F1, see Table 6), which lies slightly, but according to the χ^2 test ($p<0.05$) significantly above the baseline (66.7). All other models perform worse than the baseline. On the side of the neural models, the only hope of achieving better results is MAD-X: For metaphor detection, it seems to work well with a learning rate of 1e-3. If a sensible way of performing hyperparameter-tuning can be found, results above 70 (F1) are possible, as can be seen in Figure 9.

To conclude: since all neural models perform considerably better than the baseline for two out of three languages (and one model, MAD-X, might perform decently for all three languages), it can be stated that the neural cross-lingual models investigated in this thesis are in general applicable to MD. This is certainly the case for languages with large amounts of pretraining data.

6.1.2 Research Question 2

There are languages where only a small amount of pretraining data is available for large language models and/or that are typologically distant from the source language. Can a non-neural classifier outperform neural models for these languages?
As we saw in the previous section, neural cross-lingual models in general do not perform well for the Latin dataset. This behaviour was expected, since mBERT was pretrained with very little data on Latin and Latin is typologically dissimilar to English. Therefore we investigated whether or not non-neural models that rely on conceptual features are better suited for the Latin dataset than the neural classifiers. Indeed, the highest F1-score for the basic training dataset and default hyperparameters (see Table 6) and the augmented dataset with default hyperparameters (see Table 10) was achieved by the random forest classifier. However, the scores are rather low, with an F1-score of 72.2 being the highest score that is reached with the random forest classifier.
It can therefore be stated that the non-neural classifier did outperform the neural models with default hyperparameters for the Latin dataset, but it did so on a low level.

6.1.3 Research Question 3 and 4

It has been shown that sentences containing metaphorical language seem to be more emotionally charged than literally used sentences (Mohammad et al., 2016). Will adding emotion scores as conceptual features improve the performance of the non-neural classifier?
When using the default hyperparameters and the basic training dataset for the random forest classifier (see Table 6), VAD-scores improved the performance of the

Latin and the German classifier (by 3.9 and 0.9 points, respectively), but not the performance of the Russian classifier (it dropped by 1.9 points). When using the default hyperparameters and the augmented training dataset (see Table 10), VAD-scores improved the performance of the Russian and German classifier slightly (0.3 and 0.5 points, respectively), but not the performance of the Latin classifier (its performance dropped by more than 5 points). Therefore, no clear picture can be obtained from these results, even though there seems to be a slight tendency indicating that VAD-scores improve the performance in some cases. Maybe the picture would be clearer if VAD-scores for the entire sentence were used[39], which is suggested by the following example:

(27) Den Job kannst du dir abschminken. (translation: "You can forget about the job.")

If an emotion score was given for the whole sentence, the valence score would be rather low, because the message is unpleasant. However, each individual word per se is not unpleasant: on the contrary, the English word *job* (which is the translation of the German word "Job") has a valence rating of 0.694, which is rather positive, and *abschminken* would (individually) be translated as "remove makeup", where *remove* has a valence score of 0.292 and *makeup* of 0.74, which is on average a neutral or slightly positive score. When using the VAD-score of the entire sentence, the score would be a suitable indicator of metaphoricity: Example sentence (27) and other metaphorical uses of *abschminken* would gain a negative valence score, while literal usage as in sentence (28) would gain a neutral score.

(28) Den Eyeliner kannst du dir abschminken. (translation: "You can remove the eyeliner.")

Research has shown that mid-range abstractness ratings are unreliable, since they exhibit a large degree of disagreement among annotators (Pollock, 2018). Will separating mid-range abstractness ratings from truly concrete and abstract ratings im-

[39] In order to obtain VAD-ratings for entire sentences, a classifier would have to be trained.

prove the performance of the non-neural classifier?
In Tables 6 (results for the basic training set with default hyperparameters) and 10 (results for the augmented training set with default hyperparameters), only in one case did using three abstractness labels for the feature generation instead of two result in a stronger F1-score: for the basic training dataset and the default hyperparameters the score for Latin improved from 66.7 to 68.4 (F1). Therefore, it can be concluded that separating scores between 0.3 and 0.7 from the truly abstract scores (0.7 to 1.0) and the truly concrete scores (0.0 to 0.3) does not lead to an improved performance of the random forest classifier.[40] These results are surprising, given that the picture in previous literature (Pollock (2018), Frassinelli and Schulte im Walde (2019)) clearly indicates that mid-range ratings are not reliable. One explanation for the poor results might be that due to using 3 labels instead of 2, during training some features such as *subject_ verb_ cc* for imageability are not seen anymore (see also Section 5.1). Therefore they also cannot be used during inference, even though they might occur in the test data.

6.2 Additional Insights

6.2.1 Hyperparameter-Tuning

Neural Models: Section 6.1 did not take into account the results presented in Table 9, because the hyperparameters obtained from the grid search using the English validation dataset did not yield reliable results. As can be seen in Table 14, the correlation between the results for the different hyperparameter sets for the English validation set and the Russian, German and Latin datasets was moderate at best across all models. This means that choosing the hyperparameter set that leads to the highest F1-score on another source language dataset is not sensible, since this seems to be a case of overfitting to the source language. This is also reflected in the F1-scores shown in Table 9: the results for mBERT0 and mBERT20 are for the most part worse than the baseline, the only exception being MAD-X, which in this

[40]Trying out different different boundaries preliminarily did not result in improved F1-scores, but this should be investigated in a more structured way.

scenario shows a better performance with the hyperparameters gathered from the English validation set compared to the default hyperparameters. However, it should be taken into account that the results for the hyperparameter search on the English validation set stay within a very narrow range (see Figure 9): Especially for the first 18 indices, i.e. learning rates of 1e-3 and 1e-4, the hyperparameter sets lead to a mean F1-score of 39.4 with a standard deviation of 1.2 points. Therefore, a slight change (e.g. a different seed) could lead to different maximum F1-score on the English dataset. For example, if a change in seed led to best performance of the English validation dataset for a hyperparameter set containing a learning rate of 1e-4, a result clearly below the baseline for the target language would follow.
While the evaluation results in Section 5.6 clearly show that a target language dataset is not suitable as a validation set, they also indicate that languages other than the source language might serve well as validation sets. Figures 11 and 12 show that the correlations between English and each of the three languages Russian, German and Latin are lowest in the grid search, while the correlation for language pairs not including English is very high: the correlation between Russian and German in zero-shot classification has a value of 0.97 in terms of Spearman's ρ. Even though the correlation for the pairs Latin-Russian (0.87) and Latin-German (0.9) are lower than the correlation for Russian-German, the two correlation scores are striking, as the Latin dataset stems from a different domain than the other two datasets.
It would be interesting to investigate whether this pattern also occurs for other high- and low-level tasks, such as question answering, part of speech tagging, and so on. In this case, other languages should also be taken into account, so that it is possible to describe which languages are suited best as hyperparameter-tuning pairs.
In the current setup, the results obtained during the hyperparameter search (even the highest ones seen across all hyperparameter sets, i.e. zero-shot mBERT: 85.5 for Russian, 77.4 for German, 71.2 for Latin, MAD-X: 84.1 for Russian, 76.0 for German, 74.4 for Latin) are – with the exception of Latin – lower than the results obtained with default hyperparameters on the basic training dataset. Therefore, if more tasks and more languages are tried out, also more hyperparameters need to be taken into account, so that the hyperparameters especially for mBERT0 and mBERT20 can

actually outperform the default hyperparameters. MAD-X, on the other hand, is a promising candidate for performing MD for Latin: if we had used Russian as the validation set, we would have picked hyperparameter set 6 for the target language Latin, and this would have given us a score of 74.4 (F1) for Latin, which is the highest score achieved for Latin across all experiments. Further experiments have to confirm, though, that using data from a language different from the source and target language as validation data is a valid option.
Random Forest Classifier: Figure 10 shows that the results for using different hyperparameter sets only vary slightly and that no pattern is discernible. As the best results for the grid search overall (Russian: 80.6, German: 71.6, Latin: 68.9) are only slightly higher for German and Latin and lower for Russian, while using the augmented dataset leads to better results for the vanilla classifier in all languages, a recommendation would be to use a larger training dataset instead of performing hyperparameter-tuning for this classifier if possible.

6.2.2 Language Typology and Amount of Pretraining Data

Neural Models: Lauscher et al. (2020) showed that mBERT and other multilingual transformer models perform well on a number of tasks in languages with large amounts of pretraining data and between languages that are typologically close. Since more German than Russian data was used for pretraining mBERT (see Table 1) and since German and the source language English are typologically more similar than English and Russian, we assumed that the models would show a better performance on the German dataset than on the Russian dataset. However, this assumption was not confirmed by the experiments in this thesis.
One assumption on why models perform so well on the Russian dataset was that the German dataset is linguistically more diverse than the Russian dataset. While for the Russian dataset, the metaphoricity of the verb is clear by simply looking at the subject and/or object of the verb, this is not the case for the German dataset. Often, the verb to be classified either occurred without subject and object, or the subject or object were not suitable to make a decision about the metaphoricity of the verb, because they were pronouns, for example. Therefore, we transformed the

Latin and German datasets in such a way that they were more comparable to the Russian dataset (see Section 5.5). The more comparable datasets led to higher results for German and Latin (see Table 13) than the results from the previously used, unreduced datasets (see Table 6). However, the reduced version still shows a better model performance for Russian than for German and a better performance for German than for Latin.

As mentioned before, it is not surprising that the performance of the neural classifiers on the Latin dataset is rather weak: the language stems from the Italic branch of the Indo-European language family, whereas English belongs to the Germanic branch. This difference is marked by the fact that Latin is a synthetic language that not only relies heavily on word inflection, but also has a rather free word order, whereas English is an analytic language where hardly any inflections exist, but word order is rather fixed. In addition, the Latin pretraining data for mBERT consisted of roughly 5% of the amount of Russian or German pretraining data (see Table 1), and the domain (religion) is very different from the domain of the training data (web domain) and pretraining data (knowledge domain: Wikipedia).

Russian is also a synthetic language. English, however, contains more loan words from Latin than from Russian as well as sharing the same script with Latin. Therefore, – if language similarity alone was the deciding factor – Latin would perform better. Here it can be seen that the amount of pretraining data and the domain plays a crucial role.

However, why German (not as synthetic as Russian and Latin, same branch of the language family as English, same script, same domain) performs worse than Russian, even in a linguistically comparable dataset, cannot be explained by the results obtained so far. Therefore, a qualitative analysis of the results obtained by the best-performing neural classifier (mBERT0 with default hyperparameters and basic training dataset) for German follows in Section 7.1.

Non-Neural Models: As mentioned before, weak performance of the neural models on the Latin dataset was expected. Why the scores for Latin are so low for the non-neural model, however, is unclear: as the random forest model with conceptual features takes lemmatized subjects, verbs and objects as basis for the features, language characteristics such as word order or inflections should not influence model performance. Therefore, we performed a qualitative analysis of the results for Latin in Section 7.2.

7 Qualitative Analyses

The following sections present the results of a qualitative analysis. For that, the classification results for Latin and German were analyzed manually in order to find possible reasons for misclassification. For Russian, only a very shallow analysis was conducted due to lack of knowledge of the Russian language. For each language, the best-performing model was used to produce the results to be analyzed.

7.1 German

For the qualitative analysis, possible sources of errors were identified by looking at the predictions of the best performing model for German, namely the zero-shot mBERT with default hyperparameters and basic training dataset.
One hypothesis as to why the models perform unexpectedly mediocre on the German dataset is that the target words consist of "computationally challenging" (Köper and Schulte im Walde, 2016) **particle verbs**: they consist of a base verb (e.g. *schminken*) and a particle (e.g. *ab-*), they are highly productive and often ambiguous (Köper and Schulte im Walde, 2016), and the particle is in many syntactical constructions separated from the base verb, as in the following sentence:

(29) Der Pfarrer legte seinen ernsten Gesichtsabdruck ab. (translation: "The vicar removed his serious expression.")

If the language model computes a representation for the [MASK]-token in

(30) "Der Pfarrer [MASK] seinen ernsten Gesichtsabdruck ab." (translation: "The vicar [MASK] his serious expression."),

the gap is probably filled with a representation close to *legte*. Afterall, the gap needs to be filled with another particle verb with the particle *ab-* and there are hardly any other words that fit here (except for maybe *abändern*). The representation calculated for the masked token is therefore probably very close to the original word, and this leads to a high similarity of the representations of the masked and unmasked sentence. A high similarity in turn leads the model to classify that both sentences appear in the same context, so it falsely classifies the target word as literal. To compare: in the English translation of the sentence, *removed* can be replaced by many alternatives, such as *lost*, *altered*, or even *noticed* or *dismissed*, because it is not constricted by a particle. Therefore, the representation of the [MASK]-token would not be as close to the representation of the target word as in German. Russian also does not have particle verbs, so it would behave similarly to English and therefore be easier to classify for MD[41].

As the particle verbs consist of a base verb and a particle, the particle verbs are tokenized into at least two subtokens, while the English target verbs for the most part consist of only one subtoken. Therefore, the assumption was that the number of subtokens may be one reason why the models perform worse for the German dataset than for the Russian one. However, the mean number of subtokens for the German dataset is 3.1, while for the Russian one it is only slightly lower (2.9; for English: 1.49). The richness of Russian morphology and inflection seems to outweigh the fact that each target word in the German dataset consists of one extra subtoken for the particle. Since both languages show similar numbers of subtokens for the target word, the number of subtokens is not a deciding factor for misclassifications.

Among the misclassified test instances were 11 instances where **unusual proper nouns** preceded or followed the target words within a window of 2 words. Here are two examples[42]:

(31) Wir gedachten, Euch am Mückenflusse anzutreffen. (translation: "We planned

[41]In order to confirm the hypothesis that target words with separated particles (e.g. *legte ab*) are harder to classify than target words that consist of only one word (e.g. *ablegen*), the dataset needs to be reduced to such instances where no separations occur. This will be left for future work.

[42]A list of all misclassified instances with unusual proper nouns can be found in Appendix A.10.

to meet you at the Mosquito River.")

(32) Möchte dem Fahrzeug einen GP Geniussport 80 R mit Speed 12 T einpflanzen. (translation: "Want to implant a GP Geniussport 80 R with Speed 12 T into the vehicle.")

In order to find out whether the unusual proper nouns cause misclassifications, the proper nouns were replaced by common nouns (sentence 1: *Mückenfluss* became *Fluss*, sentence 2: *GP Geniussport 80 R mit Speed 12 T* became *einen neuen Motor*) and then the classification with zero-shot mBERT was repeated (for the alterations on all eleven instances see Appendix A.10). With the original unusual proper nouns the F1-score was 0.0, as no instance was classified correctly. For the instances with the replacements, an F1-score of 53.3 was reached, because 4 out of 11 instances were classified correctly. This is a slight indication that unusual proper nouns might cause errors. It could be that they distort the representation of the masked token, since the language model has never seen these nouns before and therefore it cannot decide which word might follow or precede (i.e. which representation is valid for the [MASK]-token).
Another hypothesis as to why the models perform worse on the German dataset than on the Russian dataset is that the German dataset contains many **idioms**. For example:

(33) Da wird der Teufel mit dem Beelzebub ausgetrieben. (translation: "One evil is replaced by another.")

Interestingly, highly similar variants of this idiom are classified inconsistently: While the target word in sentence (33) was misclassified as being literal, it was classified as non-literal in the following sentence:

(34) Denn die Elite und die USA werden den Teufel nicht mit einem Beelzebub austreiben. (translation: "For the elites and the U.S.A. will not replace one evil with another.")

In total, 3 out of 7 sentences that contain the idiom *den Teufel mit dem Beelzebub austreiben* were classified incorrectly.
Other examples for idiomatic verb usage or highly conventionalized expressions with inconsistent classification results are the following:

(35) Dampf ablassen (translation: "release steam")

→ 5 classified correctly, 4 incorrectly

(36) Sendung ausstrahlen (translation: "broadcast a show")

→ 6 classified correctly, 15 incorrectly

If the language model was not performing MD, but filling the gap that the [MASK]-token leaves, it would fill the gap with a word that often appears together with the words of the context. Therefore, in example sentence (33), the gap is filled with *ausgetrieben*, because this verb is a constituent of the idiom. It could therefore be expected that all idioms are classified as literal. However, the model sometimes classifies the instance as literal and sometimes it does not, so it can differentiate idioms from clearly non-literal word usage, but not reliably. To test whether the classifier indeed struggles with classifying idioms, the dataset from Ehren et al. (2020) was used for comparison. It is a dataset that consists of sentences from 34 preselected verbal idioms. For each idiom the information is given whether the idiom is used literally or non-literally; the details of the dataset and its preprocessing is described in detail in Appendix A.8. If the classifiers perform poorly on this dataset, it can be concluded that classifying idioms is difficult for our models. Therefore, all neural models were tried out on the dataset by Ehren et al. (2020). In order to make this dataset comparable to the dataset by Köper and Schulte im Walde (2016), it was balanced and reduced to 2000 instances. The result for the different neural models is found in Table 15[43]. For mBERT0, we used the basic dataset and default hyper-

[43]The non-neural models were not tried out because subject and object are not marked in the dataset and because the main question that we are trying to answer is why the neural models perform worse for German than for Russian.

	German (Ehren)	German (Koeper)
majority vote	0.67	0.67
mBERT0	72.1	77.6
mBERT20	76.2	76.6
MAD-X	66.4	75.1

Table 15: F1-scores for detecting non-literal usages in verbal idioms in the dataset by Ehren et al. (2020) using the default hyperparameters and basic training set for zero-shot classification with mBERT and default hyperparameters with the augmented training dataset for few-shot mBERT and the MAD-X classifier. For comparison, the F1-scores for the dataset from Köper and Schulte im Walde (2016) from the corresponding experiments are given in gray.

parameters. For mBERT20 and MAD-X, we used default hyperparameters as well, but with the augmented training dataset, because using the basic training dataset led to results below the baseline for the original German dataset (see Table 6).

As can be seen in Table 15, across all models the results for the dataset by Ehren et al. (2020) are lower than the results for the dataset by Köper and Schulte im Walde (2016). This slightly indicates that the neural methods for word-based metaphor detection do not work as well on idioms as they do on less conventionalized metaphors, especially since the verbs to be classified as literal or non-literal here (*liegen*, *stehen*, *ziehen*, etc.) are less computationally challenging than the particle verbs in the dataset by Köper and Schulte im Walde (2016).

Another weakness of the classifier seem to be instances, where the target verb is part of a larger, **extensively described metaphor**, as in the following 4 sentences:

(37) Das Volk wird gemolken, ja der letzte Tropfen wird noch ausgesaugt. (translation: "The people are milked, even the last drop is sucked out of them.")

(38) Aber ob man der Lufthansa so das Wasser abgräbt? (translation: "But does this take the bread out of Lufthansa's mouth?")

(39) In der Gerüchteküche wurde tagelang deftig aufgekocht. (translation: “For days the gossip factory was working overtime.”)

(40) Während die neuen Gegner Chris Pine und Christoph Waltz den Jungs manche Sprosse der Karriereleiter absägen. (translation: “While the new opponents Chris Pine and Christoph Waltz saw the rung off the boys' career ladder.”)

All 4 sentences have in common that not only the target word is used metaphorically, but most words of the sentence. For example, in sentence (39), only the partial word *Gerüchte-* determines that a metaphor is used here, while all other words belong to a richly illustrated metaphor. Also, all 4 sentences were classified as literal with the gold label being non-literal. Apparently, too little evidence hinted at the non-literal verb usage in these sentences.
From 1792 sentences in the balanced dataset that we used for the experiments in this thesis, 398 were misclassified. Potential sources of errors were unusual nouns (responsible for 11 misclassifications), idioms (responsible for 22 misclassifications) extensively described metaphor (responsible for 4 misclassifications). All in all, only for 37 out of 398 misclassifications, a possible explanation was found. This means that the vast majority of instances were misclassified either due to the structural difficulty of particle verbs or the reasons for the misclassifications still have to be found.
This qualitative analysis hints at some difficulties (i.e. difficulty of particle verbs, unusual proper nouns, extensively described metaphors, idioms) which might make it harder for the models to classify the German dataset than the Russian dataset, however, it has to be analyzed by a Russian speaker whether these difficulties are not found in the Russian dataset, too.

7.2 Latin

It is clear why the mBERT-based models do not perform well for Latin: pretraining data for mBERT was very limited, Latin stems from a different branch of the Indo-

European language family than the source language English, and the dataset covers a different domain than the pretraining and training data for mBERT. However, the random forest classifier is language agnostic (Tsvetkov et al., 2014), so it should perform clearly better than the baseline. Whereas for German in many cases it is not clear why misclassifications happened, for Latin in most cases it is clear. Out of the 100 instances in the dataset, 37 were misclassified. Only for 10 of the misclassifications, no obvious source for the error was found. All other misclassifications happend because of one of the following reasons:

- The most common source for misclassification was that there was no subject or object dependent on the target verb. Therefore, features could only be extracted for the verb, which does not yield enough clues for the classifier to decide whether verb usage is literal- or non-literal (17 sentences).
- Secondly, the clues that hinted at metaphoricity were found beyond the subject or object (6 sentences).
- Thirdly, for 4 sentences individual reasons for the errors were found:

  (41) Consepulti enim sumus cum illo per baptismum in mortem: ut quomodo Christus **surrexit** a mortuis per gloriam Patris, ita et nos in novitate vitæ ambulemus. (translation from American Standard Version in Logos: We were buried therefore with him through baptism into death: that like as Christ **was raised** from the dead through the glory of the Father, so we also might walk in newness of life.)

  → The word *surgere* can mean "to get up" or – in this case – "to rise from the dead". Without knowing further context, the classifier could not disambiguate the word meaning here.

  (42) **Relinque** ibi munus tuum ante altare, et vade prius reconciliari fratri tuo: et tunc veniens offeres munus tuum. (translation from American Standard Version in Logos: **Leave** there thy gift before the altar, and

go thy way, first be reconciled to thy brother, and then come and offer thy gift.)

→ The word *munus* has multiple meanings, the majority of them being abstract: gift, function, present, service, spectacle, duty, task, burden, favour. Therefore, averaging the features for the different translations probably led to a representation of the word that was rather abstract, which is why the classifier labelled it as non-literal instead of literal.

(43) Et ego dico tibi, quia tu es Petrus, et super hanc petram **aedificabo** ecclesiam meam, et portæ inferi non praevalebunt adversus eam. (translation from American Standard Version in Logos: "And I also say unto thee, that thou art Peter, and upon this rock I will **build** my church; and the gates of Hades shall not prevail against it.")

→ *Ecclesia* is not meant literally here, but in the sense of church community. The classifier used it as a concrete word.

(44) Petrus vero ad illos: Poenitentiam, inquit, agite, et baptizetur unusquisque vestrum in nomine Jesu Christi in remissionem peccatorum vestrorum: et **accipietis** donum Spiritus Sancti. (translation from American Standard Version in Logos: "And Peter said unto them, Repent ye, and be baptized every one of you in the name of Jesus Christ unto the remission of your sins; and ye shall **receive** the gift of the Holy Spirit.")

→ The word *donum* is meant figuratively here, even though it is rated as concrete.

- As mentioned before, for 10 sentences no obvious reason for the misclassification was found. The sentences are listed in Appendix A.10.2.

Removing the first and second source for errors led to an F1-score of 73.3 instead

of 72.2 (see Table 13). The improvement made in the F1-score is little because the dataset only consisted of 50 instances afterwards; the instances with errors due to individual reasons and the errors where no source could be found made up a larger percentage.
All in all it can be stated that the random forest classifier with conceptual features shows great potential for Latin datasets where the metaphoricity of the target word depends only on the (existing) dependent subject or object, as is the case for the Russian dataset.

7.3 Russian

For Russian, the results achieved with mBERT0 with default hyperparameters and augmented training data were very high (F1: 90.5), so very few errors were produced. These errors could not be analyzed due to a lack of knowledge of the Russian language. One hypothesis as to why the models perform so well on Russian is that the sentences in the Russian dataset were the shortest with an average of 9 tokens per sentence. The average sentence length for the German dataset is 13 tokens, and for Latin it is 22 tokens (see Section 3).

8 Conclusion

While the focus of previous research has been on MD in English, this thesis has shown that MD can also be successfully carried out cross-lingually. The experiments confirmed that the cross-lingual neural models mBERT0, mBERT20 and MAD-X do not only yield state-of-the-art results for standard tasks such as question answering or part-of-speech tagging, but also for MD (see research question 1).
As expected, the neural classifiers performed well on the target languages Russian and German, but not on Latin. This is mainly due to the fact that mBERT, which is the basis for all neural models used in this thesis, was pretrained on comparably little Latin data. However, the non-neural random forest classifier with vector space

model and conceptual features was able to outperform the neural models with default hyperparameters for Latin, even though the results were still at a low level (see research question 2).
Adding emotion features and treating mid-range abstractness and imageability ratings differently from the more reliable concrete and abstract ratings did not – in contrast to what was expected – boost the performance of the original classifier by Tsvetkov et al. (2014) (see research questions 3 and 4).
Even though the neural models outperformed the baseline for both the Russian and the German dataset, it is not clear why across all experiments they performed better for Russian than for German. A qualitative analysis revealed that this might be due to the inherent difficulty of particle verbs, unusual proper nouns, extensively described metaphors, and idioms that occur in the German dataset. Also, the fact that the Russian sentences are generally shorter than the German ones probably boosts performance on the Russian dataset. However, for the majority of misclassifcations no obvious reason could be found; here, XAI methods like SHAP (Lundberg and Lee, 2017) would be beneficial in order to find out why the models perform worse on the German dataset than on the Russian one.
Finally, it was shown that in most cases the misclassifications of the random forest classifier in the Latin dataset could be explained (they are mostly due to the challenging linguistic nature of the Latin dataset, where subjects and objects are missing or where the metaphoricity of the target word is determined by words beyond the subject and object). This speaks in favor of the random forest classifier, which would probably perform better on a Latin dataset that is linguistically more comparable to the Russian dataset.
When trying to find the optimal model for each of the target languages, a difficulty had to be overcome: hyperparameter-tuning seemed to be needed for good results (especially for MAD-X), however, no validation data for the target languages was available. Therefore it was tried out whether performing hyperparameter-tuning on source language material produced decent results. This procedure seems to lead to overfitting on the source language; however, it was found out that using a dataset from a language different from the source and target language (i.e. from a third

language) as validation data can lead to finding effective hyperparameters for the target language.
For future application of the models, the best performing model for each of the target languages is listed as follows:

- For Russian, zero-shot mBERT with default hyperparameters and an augmented training dataset worked best.
- For German, zero-shot mBERT with default hyperparameters and the basic training dataset worked best. However, if more is found out about which shots lead to good results, few-shot mBERT with the basic training dataset might be able to outperform zero-shot mBERT.
- For Latin, the random forest classifier with vector space representations and conceptual features worked best. However, – if a suitable validation dataset is available – MAD-X is also a promising candidate for MD classification in Latin. If a successful hyperparameter set is found, MAD-X performs well on Latin. Then, this neural approach outperforms the random forest classifier and it is also more efficient, as the preprocessing for the random forest classifier (i.e. finding lemmatized subjects, verbs and objects) is laborious, and finding a dictionary that yields sensible one-to-many translations is difficult.

9 Future Work

In future work it would be interesting to compare the performance of the previously described VSM used for the random forest classifier to a method by Vulić and Moens (2013), which bootstraps a bilingual vector space based on two monolingual non-parallel corpora (see Section 2.2). One advantage of this approach is that there is no need to use bilingual dictionaries to transfer, for example, supersense- and abstractness-ratings during inference, which is beneficial because one-to-many translations are hard to find, especially for low-resource languages, which benefit most from the non-neural vsm models (see Appendix A.5).

Different shots for mBERT 20 led to vastly different results. There should be an analysis of which factors determine whether or not a shot improves the performance of the model compared to zero-shot classification.
In order to avoid an analysis of the shot-selection and also to avoid fine-tuning hyperparameters, a strategy by Schmidt et al. (2022) could be attempted: the authors propose to replace few-shot classification with mBERT by joint fine-tuning on source and target language data, where the loss L is defined as a weighted sum of the the source and target language losses:

$$L = \delta L_{source} + (1 - \delta) L_{target} \tag{4}$$

Here, the hyperparameter δ determines “the relative weight between the two losses”. They show that this procedure leads to more robust models that are not in need of shot-selection or hyperparameter-tuning (at least for the tasks they considered, for example for part-of-speech tagging and natural language inference).
For scenarios where only zero-shot classification is possible, because absolutely no additional training material is available, it would be interesting to gain deeper insights into hyperparameter-tuning using data from a language different from the source and target language as validation data. More tasks should be tried out, and the suitability of language pairs in this setup should also be explored. These insights could prove fruitful especially for MAD-X, where the only well-performing hyperparameters for Latin could be found with the help of hyperparameter-tuning.
As mentioned before, to understand why German performed worse than Russian across all experiments, the neural blackbox models should be made more transparent with the help of post-hoc explanations such as SHAP (Lundberg and Lee, 2017).

References

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.125. URL `https://aclanthology.org/2022.acl-long.125`.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596516495.

Julia Birke and Anoop Sarkar. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 329–336, Trento, Italy, April 2006. Association for Computational Linguistics. URL `https://aclanthology.org/E06-1042`.

Yo Joong Choe, Kyubyong Park, and Dongwoo Kim. word2word: A collection of bilingual lexicons for 3,564 language pairs. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3036–3045, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL `https://aclanthology.org/2020.lrec-1.371`.

Verna Dankers, Marek Rei, Martha Lewis, and Ekaterina Shutova. Modelling the interplay of metaphor and emotion through multitask learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2218–2229, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1227. URL `https://aclanthology.org/D19-1227`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Pro-*

ceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Erik-Lân Do Dinh, Steffen Eger, and Iryna Gurevych. One size fits all? A simple LSTM for non-literal token and construction-level classification. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 70–80, Santa Fe, New Mexico, August 2018. Association for Computational Linguistics. URL `https://aclanthology.org/W18-4508`.

Rafael Ehren, Timm Lichte, Laura Kallmeyer, and Jakub Waszczuk. Supervised disambiguation of German verbal idioms with a BiLSTM architecture. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 211–220, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.figlang-1.29. URL `https://aclanthology.org/2020.figlang-1.29`.

Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/E14-1049. URL `https://aclanthology.org/E14-1049`.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database.* Bradford Books, 1998.

Diego Frassinelli and Sabine Schulte im Walde. Distributional interaction of concreteness and abstractness in verb–noun subcategorisation. In *Proceedings of the 13th International Conference on Computational Semantics - Short Papers*, pages 38–43, Gothenburg, Sweden, May 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-0506. URL `https://aclanthology.org/W19-0506`.

Maximilian Köper and Sabine Schulte im Walde. Distinguishing literal and non-literal usage of German particle verbs. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 353–362, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1039. URL `https://aclanthology.org/N16-1039`.

Zoltan Kövecses. *Metaphor: A Practical Introduction.* Oxford University Press, 2010. ISBN 9780199705313.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.363. URL `https://aclanthology.org/2020.emnlp-main.363`.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf`.

Weicheng Ma, Ruibo Liu, Lili Wang, and Soroush Vosoughi. Improvements and extensions on metaphor detection. In *Proceedings of the 1st Workshop on Understanding Implicit and Underspecified Language*, pages 33–42, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.unimplicit-1.5. URL `https://aclanthology.org/2021.unimplicit-1.5`.

Saif Mohammad. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 174–

184, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1017. URL `https://aclanthology.org/P18-1017`.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/S16-2003. URL `https://aclanthology.org/S16-2003`.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online, October 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.7. URL `https://aclanthology.org/2020.emnlp-demos.7`.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.617. URL `https://aclanthology.org/2020.emnlp-main.617`.

Lewis Pollock. Statistical and methodological problems with concreteness and other semantic variables: A list memory experiment case study. *Behavior Research Methods*, 50:1198 – 1216, 2018.

Lior Rokach and Oded Maimon. Data mining with decision trees - theory and applications. 2nd edition. In *Series in Machine Perception and Artificial Intelligence*, 2014.

Fabian David Schmidt, Ivan Vulić, and Goran Glavaš. Don't stop fine-tuning: On training regimes for few-shot cross-lingual transfer with multilingual language models. In *Proceedings of the 2022 Conference on Empirical Methods* *in Natural Language Processing*, pages 10725–10742, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.736.

Roland Schäfer and Felix Bildhauer. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet UÄŸur DoÄŸan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 486–493, Istanbul, Turkey, 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7. URL http://rolandschaefer.net/?p=70.

Ekaterina Shutova and Simone Teufel. Metaphor corpus annotated for source - target domain mappings. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010. European Language Resources Association (ELRA).

Yulia Tsvetkov, Elena Mukomel, and Anatole Gershman. Cross-lingual metaphor detection using common semantic features. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 45–51, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-0906.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 248–258, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1024. URL https://aclanthology.org/P14-1024.

Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages

680–690, Edinburgh, Scotland, UK., jul 2011. Association for Computational Linguistics. URL `https://aclanthology.org/D11-1063`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Ivan Vulić and Marie-Francine Moens. A study on bootstrapping bilingual vector spaces from non-parallel data (and nothing else). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1613–1624, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `https://aclanthology.org/D13-1168`.

Joshua R. Westbury, Kris Lyle, Jimmy Parks, and Jeremy Thompson. *The Lexham Figurative Language of the Bible Glossary.* Lexham Press, 2016.

Michael Wilson. Mrc psycholinguistic database: Machine-usable dictionary, version 2.00. *Behav. Res. Methods Instrum. Comput.*, 20(1):6–10, January 1988.

A Appendix

A.1 Abbreviations

The following abbreviations are used in this thesis:

MD	metaphor detection
mBERT0	zero-shot multilingual BERT
mBERT20	few-shot multilingual BERT (target language fine-tuning with 20 instances taken from the test data)
VAD	valence, arousal, dominance
en	English
ge	German
ru	Russian
abstr_imag_3	abstract, neutral, concrete (Ginitract, concrete)
VSM	vector space model
MLM	masked language modelling
NSP	next sentence prediction

Table 16: Abbreviations used in the thesis.

A.2 Terminology: *Metaphorical*, *Non-Literal* and *Figurative*

Table 17 shows that the terms *metaphorical*, *non-literal* and *figurative* are used interchangeably in the papers that describe the datasets that we use in this thesis: either the papers avoid the term metaphorical and use the broader terms *non-literal* or *figurative*, or they use the term *metaphorical*, but mention that this term is not clear-cut and/or list instances which are not metaphors (but, for example, personifications) in their dataset. Originally, the focus of this thesis was supposed to be on metaphors, but as the different datasets contain other forms of figurative language, too, classification using the given datasets always involves other forms of figurative language as well. We therefore also use the terms *non-literal* and *figurative* in addition to *metaphorical*.

dataset	**terminology**	**comments**
Tsvetkov et al. (2013)	metaphorical	They mention in the paper that "there is no clear-cut semantic distinction between figurative and metaphorical language." They also use personifications in their dataset (see example sentence (2)).
Köper and Schulte im Walde (2016)	non-literal	
Lexham by Westbury et al. (2016)	figurative	They mention that they include metaphors, similes and metonymies, "as . well as other types of non-literal expressions."
Mohammad et al. (2016)	metaphorical	The dataset includes figurative language that might be interpreted not as a metaphor, but, for example, as a personification (as in: "his high fever attested to his illness").

Table 17: Terminology used in the different datasets.

A.3 Reproducibility of Results

The results presentend in this thesis were obtained by using the GPUs on the servers *froschweihe* and *strauss*. If the experiments are repeated on the same server, the results are entirely reproducible. However, if the experiments that were conducted on *froschweihe* are repeated on *strauss*, the results slightly differ, even if the same hyperparameters and the same seed is used. Therefore it is to be expected that slight changes also occur when the code is run on other servers.
Since some experiments have been carried out on *froschweihe* and some on *strauss*, the computing infrastructure of both servers is given here:

- *strauss*: The server contains an AMD EPYC 7542 32-Core Processor with 64 threads and NVIDIA GeForce RTX 2080 Ti GPUs.
- *froschweihe*: The server contains an AMD EPYC 7282 16-Core Processor with 32 threads and NVIDIA RTX A6000 GPUs.

A.4 WordPiece

Word Piece is an algorithm that splits an input string into subword tokens from a predefined vocabulary. In order to arrive at such a vocabulary of subword tokens, the following procedure is carried out: As a first step, the vocabulary is initialized with individual characters found in the training data. These are used as the first tokens. Then the following steps are carried out until the desired vocabulary size is reached:

1. A score is computed for each token pair in the vocabulary with the following formula (p stands for "part"):

   (5) $score = (freq\_of\_pair)/(freq\_of\_first\_p \times freq\_of\_second\_p)$

2. The pair with the highest score is merged and added to the vocabulary.

In the formula, the frequency of a pair (e.g. *unable*) is divided by the product of its parts, namely the product of the frequency of *un* and the frequency of *##able*. Even though *unable* occurs frequently in the vocabulary, it's individual parts appear very often in other combinations, too (e.g. in *unholy*). Therefore the word is not merged. Only a word where its parts do not often appear individually is merged, as for example with the word *computer*, where *com* might appear quite frequently, but *puter* does not.

A.5 Bilingual Dictionaries

The options presented in Table 18 were tried out to find an electronic bilingual dictionary, which produces one-to-many translations (for example: *Fliege* – "fly, bowtie"). The translations produced by Word2Word are not perfect, as can be seen from the following example:

(45) abschminken – 'makeup', 'make-up', 'remove', 'goodbye', 'kiss'

On the one hand, no direct translation of abschminken was given, the two words *remove* and *makeup* are presented separately. On the other hand, unrelated words like *kiss* and *goodbye* are listed as translations. There are translations, that work rather well, for example:

(46) Vorsatz – 'resolution', 'premeditation', 'intent', 'premeditated', 'intention'

Yet even in this sentence the word *premeditated* is not a correct translation. Also, many translations are missing, for example the word *Zecke* ("tick").
However, Word2Word was used in the thesis because it produces the best results for our purposes among the different solutions that were tried out.
In addition to these options, there are scrapers available as python packages, but they were not considered because their output is illegal.

	Advantages	**Disadvantages**
Oxford Dictionary API	python-friendly (API via requests module in Python)	No Latin, no translations to individual words, but translations of entire example sentences.
DeepL API	Python-friendly (API via requests module in Python)	One-to-one translations only.
Google Translate API	Python-friendly (API via requests module in Python)	One-to-one translations only.
Pons API	one-to-many translations	Not python-friendly, translations are mixed with example sentences.
WordReference API		API keys are no longer handed out, so it stopped working.
Marian / FairSeq	Can in theory produce the top k translations for a word.	Does not work well without context.
Word2Word	one-to-many translations	Translations are often not sensible: *книга* is translated as 'book', 'Book', 'favorite', 'open', 'read', even though only 'book' is correct.
dict.cc	Easy to download, mostly one-to-many translations, available for Latin	Translations are mixed with example sentences, very hard to preprocess. For Latin, highly frequent words are missing and there are mostly one-to-one-translations.

Table 18: Table presenting different options that were tried out in order to find a dictionary that produces one-to-many translations.

A.6 Hyperparameter Sets

A.6.1 mBERT0 and mBERT20: Index to Hyperparameter Set Mapping

During the hyperparameter search described in Section 5.2, the sets of hyperparameters listed in Table 19 were used for mBERT0 and mBERT20:

index	**learning rate**	**epochs**	**train batch size**
1	1e-4	8	8
2	1e-4	8	16
3	1e-4	8	32
4	1e-4	16	8
5	1e-4	16	16
6	1e-4	16	32
7	1e-4	32	8
8	1e-4	32	16
9	1e-4	32	32
10	1e-5	8	8
11	1e-5	8	16
12	1e-5	8	32
13	1e-5	16	8
14	1e-5	16	16
15	1e-5	16	32
16	1e-5	32	8
17	1e-5	32	16
18	1e-5	32	32
19	1e-6	8	8
20	1e-6	8	16
21	1e-6	8	32
22	1e-6	16	8
23	1e-6	16	16
24	1e-6	16	32
25	1e-6	32	8
26	1e-6	32	16
27	1e-6	32	32

Table 19: Index to hyperparameter set mapping for zero- and few-shot mBERT.

A.6.2 MAD-X: Index to Hyperparameter Set Mapping

During the hyperparameter search described in Section 5.2, the sets of hyperparameters listed in Table 20 were used for the MAD-X-classifier:

index	**learning rate**	**epochs**	**train batch size**
1	1e-3	10	8
2	1e-3	10	16
3	1e-3	10	32
4	1e-3	50	8
5	1e-3	50	16
6	1e-3	50	32
7	1e-3	100	8
8	1e-3	100	16
9	1e-3	100	32
10	1e-4	10	8
11	1e-4	10	16
12	1e-4	10	32
13	1e-4	50	8
14	1e-4	50	16
15	1e-4	50	32
16	1e-4	100	8
17	1e-4	100	16
18	1e-4	100	32
19	1e-5	10	8
20	1e-5	10	16
21	1e-5	10	32
22	1e-5	50	8
23	1e-5	50	16
24	1e-5	50	32
25	1e-5	100	8
26	1e-5	100	16
27	1e-5	100	32

Table 20: Index to hyperparameter set mapping for MAD-X.

A.6.3 RF: Index to Hyperparameter Set Mapping

During the hyperparameter search described in Section 5.2, the sets of hyperparameters listed in Table 21 were used for the random forest classifier:

index	**# of estimators**	**max depth**	**random state**
1	10	5	83
2	10	5	297
3	10	5	1234
4	10	10	83
5	10	10	297
6	10	10	1234
7	10	None	83
8	10	None	297
9	10	None	1234
10	50	5	83
11	50	5	297
12	50	5	1234
13	50	10	83
14	50	10	297
15	50	10	1234
16	50	None	83
17	50	None	297
18	50	None	1234
19	100	5	83
20	100	5	297
21	100	5	1234
22	100	10	83
23	100	10	297
24	100	10	1234
25	100	None	83
26	100	None	297
27	100	None	1234

Table 21: Index to hyperparameter set mapping for random forest.

A.7 English Validation Set

A.7.1 Dataset

When carrying out the grid search described in Section 5.2, a second source language dataset next to the dataset by Tsvetkov et al. (2014) was needed. For this purpose, the metaphor detection dataset provided by Mohammad et al. (2016) was used[44]. It consists of 1639 sentences, of which 410 are annotated as metaphorical and 1229 are annotated as literal, so the dataset is not balanced. The sentences are extracted from WordNet. Here are two example sentences:

(47) This young man knows how to **climb** the social ladder. → *non-literal*

(48) Did you ever **climb** up the hill behind your house? → *literal*

Subject, verb and object are not given as lemmatized forms. The mean sentence length is 7.4 words with a standard deviation of 2.65 words.

A.7.2 Preprocessing

As the dataset includes no subjects, verbs and objects as lemmatized forms, these were added for 100 sentences manually in order to perform the grid search for the **random forest classifier**. We carried out the same preprocessing that was described for the English dataset in Section 3.1 for this dataset (i.e. we replaced the target word with the mask-token and preprocessed the original sentence and its masked copy with the HuggingFace tokenizer pipeline) for the **mBERT-based classifiers**.

[44]`http://saifmohammad.com/WebPages/metaphor.html`

A.8 German Dataset for Idiom Detection

A.8.1 Dataset

The German dataset by Ehren et al. (2020) contains sentences from 34 preselected verbal idioms, which occur in literal and metaphorical idiomatic occurrences[45]. The dataset comprises 6985 sentences, of which 1527 are annotated as literal, 5417 as idiomatic and 8 as both idiomatic and literal. Also, for 33 sentences the annotators labelled the instance as "undecided", because they thought the context was not sufficient for making a decision. The sentences were taken from the German newspaper corpus TüPP-DZ, so the sentences are obtained from the news domain.
Here are two example sentences:

(49) „Auf den Arm nehmen kann ich mich alleine ,“ meinte er zusehends zornig und begann eine heftige Befragung. (translation: " 'I don't need you to fool me', he said increasingly angry and began a heavy interrogation.") → *non-literal*

(50) Der kleine Peter schaut verängstigt, die Mutter läßt das Telefon klingeln, nimmt ihren Sohn auf den Arm. (translation: "Little Peter looks scared, the mother lets the telephone ring, picks up her son.") → *literal*

Subject, verb and object are not given as lemmatized forms. For the experiments, we balanced and reduced this dataset to 2000 instances that do not contain the labels "undecided" and "both". The mean sentence length in our balanced dataset is 26 words with a standard deviation of 14.1 words.

A.8.2 Preprocessing

This dataset was not used for classification with the **random forest classifier**, therefore subject, object and verb were not annotated. For the **mBERT-based**

[45]`https://github.com/rafehr/COLF-VID`

models, the same preprocessing that was described for the English dataset in Section 3.1 was carried out for this dataset (i.e. replacing the target word with the mask-token and preprocessing the original sentence and its masked copy with the HuggingFace tokenizer pipeline).

A.9 Hyperparameter-Selection for Tuning

The hyperparameters for the grid search were chosen by looking at the default hyperparamters and varying them: For mBERT the default hyperparameters are a learning rate of 5e-5, 3 epochs and a batch size of 8. For the learning rate roughly the neighbouring values were chosen in addition to 1e-5, which are 1e-4 and 1e-6. Preliminary experiments also reveiled that smaller and larger learning rates lead to random results. As 3 epochs are few, larger numbers, i.e. 8, 16 and 32 epochs, were chosen. As batch-size commonly 8, 16 or 32 are tried out.
For MAD-X the default learning rate was 1e-4, so the neighbouring values 1e-3, 1e-4 and 1e-5 were chosen. As the default number of training epochs (100 epochs) was very high, smaller numbers of epochs were tried out, too (10, 50, 100). The common batch-sizes of 8, 16 and 32 were used here, too.
For the random forest classifier, the default values of 100 trees in the forest (i.e. the number of estimators) was varied so that 10, 50 and 100 trees were tried out. The default max depth of None (which means that the nodes are expanded until all leaves are pure) was varied in such a way that in addition to None also a max depth of 5 and 10 was tried out. Finally, the three random states (83, 297 and 1234) were chosen randomly to see which impact different seeds have.

A.10 Error Analysis

A.10.1 Unusual Proper Nouns

sentence	**replacement**
Statt eines Feinschmecker-Menüs zum Abschluss tischt **BioWare** bewährte Kost auf.	das Restaurant
Eine satte Portion Geschichten bekommen Museumsgäste am Nachmittag von **Maria Kiener** aufgetischt.	der Leiterin
Denn Roth hat aus der allzeit perfekten Grammatiksystematik von **Panini** abgeschrieben.	von einem Wissenschaftler
Schliesslich entschied man sich dafür zunächst in **Babylon** auszugraben.	in der Stadt
TMD hat jetzt die wohl älteste Demo Aufnahme von **Metallica** ausgegraben.	der Band
Später sind wir zum **Literary Walk** aufgebrochen und haben uns in Dublin umgeschaut.	Spaziergang
Möchte dem Fahrzeug einen **GP Geniussport 80 R mit Speed 12 T** einpflanzen.	neuen Motor
Die S4 wird ab Rahlstedt richtig leer und der nächste Mob steigt in **Ahrensburg** aus.	in der Stadt
Den Aufenthaltsort des Hexers konnten sie einer Dienerin **Barsoks** abpressen.	-
Wir gedachten, Euch am **Mückenflusse** anzutreffen.	Fluss
ich nehme auch immer **eyebright** zum abschminken, reinigt und pflegt gleichzeitig.	Reiniger

Table 22: Misclassified sentences, where unusual proper nouns (see words in bold) are replaced by common nouns. Replacement words in purple indicate that the instances were classified correctly after the original words were substituted by these replacements.

A.10.2 Latin

For the follwing instances no source for the error could be found:

(51) Amen, amen dico vobis, quia venit hora, et nunc est, quando mortui audient vocem Filii Dei. (translation from the American Standard Bible from Logos: “Verily, verily, I say unto you, The hour cometh, and now is, when the dead shall hear the voice of the Son of God. ”)

(52) Sed habeo adversum te, quod caritatem tuam primam reliquisti. (translation from the American Standard Bible from Logos: “But I have this against thee, that thou didst leave thy first love.”)

(53) Ecclesia quidem per totam Judaeam, et Galilaeam, et Samariam habebat pacem, et aedificabatur ambulans in timore Domini, et consolatione Sancti Spiritus replebatur. (translation from the American Standard Bible from Logos: “So the church throughout all Judaea and Galilee and Samaria had peace, being edified; and, walking in the fear of the Lord and in the comfort of the Holy Spirit, was multiplied.”)

(54) Dixit autem illis: Ubi est fides vestra? Qui timentes, mirati sunt ad invicem, dicentes: Quis putas hic est, quia et ventis, et mari imperat, et obediunt ei? (translation from the American Standard Bible from Logos: “And being afraid they marvelled, saying one to another, Who then is this, that he commandeth even the winds and the water, and they obey him?“”)

(55) per quem accepimus gratiam, et apostolatum ad obediendum fidei in omnibus gentibus pro nomine ejus (translation from the American Standard Bible from Logos: “through whom we received grace and apostleship, unto obedience of faith among all the nations, for his name’s sake”)

(56) Amen dico vobis, quicumque non acceperit regnum Dei sicut puer, non intrabit in illud. (translation from the American Standard Bible from Logos: “Verily I say unto you, Whosoever shall not receive the kingdom of God as a little child, he shall in no wise enter therein.”)

(57) Qui cum venissent, oraverunt pro ipsis ut acciperent Spiritum Sanctum (translation from the American Standard Bible from Logos: “who, when they were come down, prayed for them, that they might receive the Holy Spirit.”)

(58) Et nolite timere eos qui occidunt corpus, animam autem non possunt occidere: sed potius timete eum, qui potest et animam et corpus perdere in gehennam. (translation from the American Standard Bible from Logos: “And be not afraid of them that kill the body, but are not able to kill the soul: but rather fear him who is able to destroy both soul and body in hell.”)

(59) An putatis quia inaniter Scriptura dicat: Ad invidiam concupiscit spiritus qui habitat in vobis? (translation from the American Standard Bible from Logos: “Or think ye that the scripture speaketh in vain? Doth the spirit which he made to dwell in us long unto envying?”)

(60) qui solus habet immortalitatem, et lucem inhabitat inaccessibilem: quem nullus hominum vidit, sed nec videre potest: cui honor, et imperium sempiternum. Amen. (translation from the American Standard Bible from Logos: “who only hath immortality, dwelling in light unapproachable; whom no man hath seen, nor can see: to whom be honor and power eternal. Amen.”)