Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Korpus- und Computerlinguistik

A Neural-Network-Based Approach to Token-Level Metaphor Detection

Markus Opolka

markus.opolka@fau.de

Student registration no.: 22004202

Date: October 8, 2018

Supervisor: Prof. Dr. Stefan Evert

Contents

Noah: Do you even know what an idiom is?
Archer: Colloquial metaphor.

Heart of Archness: Part II

1 Introduction

As humans, understanding non-literal language rarely poses any problems and often times we do not even realize how deeply figurative speech is embedded in our day to day communication. Effortlessly we take in a broad range of situational context together with our world knowledge, to interpret or produce non-literal expressions. This may be in the form of regularly used hyperbole, when I tell my friends: "I am so hungry, I could eat an elephant!" or a simile, when something is "as cold as ice". These two examples of non-literal use of language are specialized types of metaphors.

At their core, metaphors allow us to refer to one thing by mentioning another by means of similarity. Often the words "like" or "as" evoke metaphorical comparison. When we describe something as "solid as a rock", we compare the properties of the object in question to the stability and durability of a rock. Other metaphorical use of language includes the before mentioned, hyperbole, in which the elephant stands in for a ridiculously large meal, or certain idioms such as: "having a short fuse", in which conceptual attributes of an explosive detonator are used to describe someone with a quick temper.

Machines, on the other hand, usually lack the ability to process all available situational context and or have limited world knowledge. This poses a difficult problem for natural language processing systems, for instance when detecting sarcasm or irony in sentiment analysis. This work will, first, provide an overview of computational methods for identifying non-literal, especially metaphorical, use of language. Secondly, the implementation of an identification method on a token-level using a neural network will be discussed.

In today's world, natural language processing (NLP) methods are ubiquitous: machine translation, dialogue systems, automatic summarizing and information extraction are just some examples that could benefit from being able to better recognize non-literal language. The growing market of virtual assistants, such as Apple's Siri, Google's Assistant or Amazon Alexa, also demonstrates that human-computer interaction via speech is no longer kubrickesque science fiction, but an everyday reality. By enabling these machines to better understand our creative use of language, we won't have to switch to a special machine register when speaking to them.

But not only computers have a need to better identify non-literal use of language. There is a growing toolbox of NLP systems to support human interpretation of literature and communication. These include amongst others: sentiment analysis, topic modeling and collocation extraction. Thus, human interpreters might also benefit from systems that are able to recognize non-literal language, acting like a hermeneutic magnifying glass to better see how a text uses metaphorical concepts to argue. This will be discussed further in the Summary section.

The following section will focus on the current linguistic theory of metaphors. Section 3 will

relate this work to prior publications in the field of metaphor detection and will give an overview of existing methodology. Section 4 will mainly focus on digital resources available for the problem at hand, providing an extensive list and details on how each resource relates to non-literal use of language. In section 5, the practical part of this work will be described in detail, including also a small introduction to machine learning and data preprocessing.

2 Theoretical definition of metaphors

Lakoff and Johnson's 1980 publication *Metaphors we live by* [21] has become an influential and invaluable theoretical foundation for linguistic research on the subject. "The essence of metaphor is understanding and experiencing one kind of thing in terms of another", they argue, and "metaphor is, in general, not based on similarity [...]. Instead, it is typically based on cross-domain correlations in our experience, which give rise to the perceived similarities between the two domains within the metaphor." This means, that the (two) concepts in a metaphor do not actually share any real similarities, but that we perceive similarity based upon our human experience. A popular example by Lakoff and Johnson is the ARGUMENT IS WAR metaphor, which is expressed in phrases such as:

- Your claims are indefensible.
- I have never won an argument.

"It is not that arguments are a subspecies if war. Arguments and war are different kinds of things [...] and the actions performed are different kinds of actions. But ARGUMENT is partially structured, understood, performed, and talked about in terms of WAR.", so Lakoff and Johnson. These metaphorical concepts, as seen here in small caps, are what they call: *conceptual metaphors*. They strictly separate between metaphorical concepts and linguistical expressions related to them. Thus, the conceptual metaphor ARGUMENT IS WAR has many linguistic realizations such as: "she attacked every weak point in my argument", "they had a violent argument", "we have conflicting positions". This may seem trivial, but by examining this observation further, the authors conclude that: most of our conceptual system is based on metaphors and that these originate from fundamental human experiences. Examples to illustrate this are: HAPPY IS UP and SAD IS DOWN.

- I'm feeling down.
- She fell into a depression.
- My spirits rose.
- You are really low these days.

This, according to Lakoff and Johnson, is based upon the physical experience of human posture. An erect posture is typically associated with a positive emotional state and vice versa. Directions such as UP and DOWN play a huge role in how we perceive the world, given that we are physical beings bound by the ubiquitous gravity of our home planet. Which is also basis

for another important category of conceptual metaphors named *container metaphors*. We (humans) regularly project being physical beings with a distinct boundary (our skin) onto other concepts, which is often indicated by words such as *in*, *into* or *out*. Examples of this are:

- Are you in the race on Sunday? (Race as container object)
- I put a lot of energy into my thesis! (Thesis as container object)
- We are out of trouble now.
- How do I get out of VIM?

An extensive list of examples for conceptual metaphors can be found by using the MetaNet project [3], which is a structured repository of conceptual metaphors developed by the International Computer Science Institute Berkeley.

Generally speaking, metaphors involve understanding one domain of experience in terms of a different domain. The LOVE AS JOURNEY metaphor illustrates this very clearly, in which a relationship might *not be going anywhere* or lovers have to *go separate ways*. Lakoff [20] writes: "[...] metaphor can be understood as a mapping (in the mathematical sense) from a source domain (in this case, journeys) to a target domain (in this case, love). The mapping is tightly structured. There are ontological correspondences, according to which entities in the domain of love [...] correspond systematically to entities in the domain of a journey [...]". Furthermore, conceptual metaphors typically employ a more abstract concept as target domain and a more concrete or physical concept as their source domain. As shown in the HAPPY IS UP metaphor, in which the abstract target concept of human emotions is explained by the more concrete source concept of physical directions. Or in the LOVE AS JOURNEY metaphor, in which the source concept of traveling is used to talk about the abstract target concept of human relationships. Hence, we can use a concrete and understandable concept to describe something more abstract. This also applies to personifications, where we attribute understandable human motivations and characteristics to non-human entities. In Unix operating systems an abstract process does not get terminated, it gets killed [1]. Or as in Lakoff and Johnson's example of INFLATION IS A PERSON, in which the inflation can *rob* us of our savings, or *give birth* to a money-minded generation.

Although being one of the most influential publications in the realm of cognitive linguistics, *Metaphors we live by* has also received some criticism. To quote Gerard Steen [29]:

> However, fundamental methodological criticism has been voiced about this approach (e.g. Vervaeke & Kennedy 1996; Ritchie 2003, 2004; Haser 2005). It is argued that the delimitation of conceptual metaphors is not sufficiently constrained to allow for the precise identification of specific linguistic items as related to them. Thus, Ritchie has suggested that argument may be just as easily a matter of chess as of war, and that the criteria for deciding which is which are unclear. Other scholars have questioned the need for postulating conceptual metaphors in the first place

[1] Unix systems use a program called "kill" to communicate with processes, which is usually used to terminate them.

(Murphy 1996, 1997; Glucksberg 2001; Jackendoff 2002; McGlone 2007). They claim that most metaphorical expressions in language may have nothing to do with thought, but are a matter of lexical semantics which can be historically explained.

In all, the identification of metaphors in human language has become a matter of controversy and different theoretical models to explain metaphorical language have emerged. Here is a brief overview of the most influential models. For further details on these models and an extensive overview, see Steen (2007) Chapter 3 [31].

Two-Domain Approach

The Two-Domain Approach by Lakoff and Johnson has already been discussed in great detail. As seen before, it is based upon two conceptual structures: a source domain, usually more concrete, and a target domain, usually more abstract. The systematic mappings between these domains are called conceptual metaphors, which consist of conceptual correspondences between elements in each domain.

Many-Space Approach

The Many-Space Approach replaces the two conceptual domains with four conceptual spaces. Two of these closely correspond to the source and target domain already seen, the other two try to describe the common ground between these domains and the emergent structure from the mapping. In order to fully capture a statement's meaning. For instance in "this surgeon is a butcher", the two-domain approach might map the source and target domains, but it is missing the implicit meaning: the surgeon is incompetent.

Class-Inclusion Approach

The Class-Inclusion Approach uses three conceptual categories to explain metaphors. Unlike the models presented before, this theory treats metaphorical statements not as implicit comparisons, but as explicit class-inclusion assertions. For example, in "my job is a jail" the target is assigned to a category which does not coincide with the one defined by the vehicle term *jail* in the literal sense. Instead, it is a superordinate category typified by the literal source which includes things that the source exemplifies (involuntary, confining, unpleasant). This metaphor vehicle can be construed as a member of various ad hoc categories.

Career of Metaphor Approach

The Career of Metaphor Approach includes many aspects of the two-domain approach, but also incorporates insights from the class-inclusion approach. Instead of having a fixed number of conceptual structures, it allows for a variable number. Only two conceptual structures are required for novel metaphors, corresponding to the already seen source and target domains. For conventional metaphors, however, three structures are required; these resemble the structures seen in the class-inclusion approach.

As seen, all of these theoretical models share the notion of at least two conceptual structures (i.e. source and target domain). Thus, many of the operational implementations for identifying metaphors are based upon these two domains. One highly influential method is the *metaphor*

identification procedure (MIP), developed by the Pragglejaz Group [15]. The Pragglejaz Group is an international collective of metaphor researchers who joined forces to examine whether it was possible to devise an explicit and precise method for canonical metaphor identification in discourse.

As many do, the MIP uses the theoretical framework of Lakoff and Johnson as basis for declaring metaphorical use of language. Furthermore, it operates on words - or rather *lexical units* - as the unit of analysis. A lexical unit includes all words provided with an independent part-of-speech tag, as well as fixed multi-word expressions, for instance: "by means of" and "of course". With these definitions, the MIP follows an algorithm-like structure [15], which is:

1. Read the entire text/discourse to establish a general understanding of the meaning.
2. Determine the lexical units in the text–discourse.
3.
   - For each lexical unit in the text, establish its meaning in context, i.e. how it applies to an entity, relation or attribute in the situation evoked by the text (contextual meaning). Take into account what comes before and after the lexical unit.
   - For each lexical unit, determine if it has a more basic contemporary meaning in other contexts than the one in the given context. For our purposes, basic meanings tend to be
     - More concrete; what they evoke is easier to imagine, see, hear, feel, smell, and taste.
     - Related to bodily action.
     - More precise (as opposed to vague).
     - Historically older

     Basic meanings are not necessarily the most frequent meanings of the lexical unit.
   - If the lexical unit has a more basic current contemporary meaning in other contexts than the given context, decide whether the contextual meaning contrasts with the basic meaning but can be understood in comparison with it.
4. If yes, mark the lexical unit as metaphorical.

The MIP was then extended in 2010 to include the identification of signals of metaphors, which are lexical signs that some form of contrast or comparison is at play (e.g. like, as, more, less), as well as instructions for handling new lexical units. This extended method was named MIPVU [29], and it was used to compile the VU Amsterdam Metaphor Corpus. This corpus is a hand-annotated resource for all metaphorical language use and will be further discussed in the Resources section.

The Pragglejaz procedure has been used for empirical extraction with high levels of agreement between independent annotators [2]. Even though being conceived as a procedure for human-

[2]The reported annotator agreement for the VUAMC is 0.84 in terms of Fleiss Kappa.

discourse analysts, it paves the way for experiments on computational methods for detection of non-literal use of language. It shows that the theoretical foundation of Lakoff and Johnson can successfully be operationalized for metaphor identification. As shown in the following section, many computational methods for identification are based on the source-target domain assumption.

3 Computational methods for metaphor identification

This section will give an overview of some of the most recent publications in the field of automated metaphor detection. These publications can be categorized by their approaches, with the core methods being: selectional preferences, lexical coherence, abstractness/concreteness ratings, word-sense disambiguation and machine learning. The following publications are categorized accordingly and the methods applied will be discussed briefly. Further details on these core categories and more evaluations can be found in [13].

Many of the resources mentioned in this section will have detailed explanations in the Resources section.

3.1 Word-sense Disambiguation

In natural language processing, word-sense disambiguation describes methods that aim to identify which sense (meaning) of a word is used in a given context. Methods based on word-sense disambiguation operate by treating metaphor identification as a disambiguation problem.

A clustering approach for nearly unsupervised recognition of non-literal language [9]

The Trope Finder (TroFi) approach by Birke et al. is a system for automatically classifying literal and non-literal usage of verbs. They adapt an existing word-sense disambiguation algorithm for literal/non-literal clustering through the redefinition of literal and non-literal as word-senses.

For this, the algorithm requires a target set, the set of sentences containing the verbs to be classified and two seed sets (literal and non-literal), which contain feature lists consisting of the stemmed nouns and verbs in a sentence. Then, similarities between the sentences in target and seed sets are calculated by using a similarity-based word-sense disambiguation algorithm. A target sentence is considered to be "attracted" to one of the seed sets containing the sentence to which it shows the highest similarity, thus being either literal or non-literal.

By using well established methods in natural language processing, Birke et al. were able to compile the TroFi Example Base dataset, which contains metaphor examples for 50 English verbs. Furthermore, this method can likely be adapted to also classify other word categories. However, the original experiment used custom hand-annotated data for evaluation of the algorithm, which leaves a custom adaptation to find its own method for validation and optimization.

3.2 Abstractness/Concreteness Ratings

Methods based on abstractness/concreteness ratings work by using computational measures to detect degrees of abstractness of words. This is based on the presupposition that metaphorical words from a source domain tent to use more concrete words than the target domain, as seen in the introduction.

Robust extraction of metaphor from novel data [32]

Based on Lakoff and Johnson's linguistic foundation of conceptual metaphors, Strzalkowski et al. try to identify source domains in a text by using topical structures and imageability scores. For this, a set of certain target domain keywords is defined *a priori*. Passages containing these keywords are then extracted, and so-called topic chains are identified within them, by linking occurrences of each noun and verb. Then, in order to narrow the pool of candidate relations (i.e. relations between a source and target domain), imageability scores for the topic chains are computed. Strzalkowski et al. hypothesize, that metaphors use highly imaginable words to convey meaning. As resource they use the MRCPD database [36] to look up imageability scores, excluding topic chains with low a score. The remaining words are considered candidate relations and are then further investigated. Using dependency parsing to unveil the syntactic structure of a sentences, only verbs that have the target concept in direct dependency path are further considered. The example given is: "navigate" and "labyrinthine", in the context of "governance". Finally, occurrences of these candidate relations are extracted from collocates in a balanced corpus, revealing unusual occurrences in the given input text.

This approach combines a heuristic method with well established corpus-based methods to identify possible source domains, without extensive annotated training data. The approach is, however, somewhat restricted by choosing a set of keywords beforehand, which may be incomplete or difficult to establish. A major advantage of this method is that is also tries to discover the underlying conceptual metaphor, by exploring the extracted collocates from a corpus.

3.3 Lexical Coherence

Methods based on lexical coherence work by detecting words that are semantically incoherent with other context words. There have been several different of these approaches in the recent years, which usually include the formation some sort of semantic signature for a given target domain.

For example, a very simple target domain signature like *economic inequality* might include all content words from articles regarding the topic, and then all word that are not included are detected. Naturally, the methods described here are more sophisticated. This section will show how semantic topic signatures are formed and applied.

Semantic signatures for example-based linguistic metaphor detection [24]

Mohler et al. also base their approach for detection on Lakoff and Johnson's conceptual

metaphors. They argue that it is possible to detect metaphorical language by creating a semantic signature for a known target domain, using *governance* as an example domain in their experiment. This - *a priori* defined target domain - is called a "domain sensitive semantic signature", since it is highly specialized depending on the desired target domain. Signatures are created by using a set of Wikipedia related to a given topic, these articles are extracted automatically utilizing web crawlers.

From this set of articles a target domain signature is built by exploiting the semantic information provided by WordNet [3], gathering all word senses for content words and transforming them into an undirected graph. After constructing this signature, it is possible to use it to map a given text (or sentence) into a multidimensional conceptual space, which allows for comparison of two texts directly based on their conceptual similarity. By mapping a set of known metaphors into this space, it is possible to calculate the likelihood that a given text contains some metaphor within the same target domain.

However, with the recent developments in the field of distributional semantics and word embeddings [4], this implementation for metaphor detection might have become obsolete. This is not to say, that the core concept cannot be adapted to include these newer methods.

Metaphor detection through term relevance [28]

Schulder et al. use a statistical approach for metaphor detection. They hypothesize that "out of place" words in a text are generally not meant literally. Thus, being able to detect words that do not match the texts typical vocabulary, means being able to detect potential metaphorical language. For this, they use term-relevance as a statistical measure for the unusualness of a word in a given context.

This term-relevance consists of two features: First, domain-relevance, which measures whether a term is typical for the literal target domain or not. Second, common-relevance, which indicates terms that are so commonly used across domains that they have no discriminative power. For domain-relevance they adopt the TF-IDF measure (term frequency inverse document frequency) [5], by treating all texts of a domain as a single document. Common-relevance as a second measure is used to filter out common words across a domain. Thus, a low term-relevance score is likely to indicate a metaphor candidate.

With a rather simple computational measure at its core, this methods appears very promising, being language and word category independent. Results from the original experiment suggest that term-relevance can also help when data is sparse. Schulder et al. propose to reimplement the underlying concept of term-relevance using different methods such as word embeddings.

Metaphor detection in discourse [16]

[3] WordNet is a lexical database for the English language created by the Princeton University

[4] Word embeddings describe techniques of representing words or phrases as vectors of real numbers. For further details on this see: Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.

[5] TF-IDF is commonly used in information retrieval and test mining. It is a statistical measure used to evaluate how important a word is to a document in a collection of documents. Put simply, the higher the TF-IDF score, the rarer the term and vice versa.

Jang et al. also identify metaphors by detecting words or phrases that are semantically incoherent within their context. They do so by using different semantic resources to first define a global (document-wide) context and a local (sentence-wide) context. After that, these pregenerated contexts are leveraged in order to detect words that are potentially metaphorical.

Global contexts are generated by using the following resources: the semantic categories for each word using FrameNet [6] A topic distribution for each word derived from a 100-topic Latent Dirichlet Allocation model [7]. And lexical chains [8] to obtain multiple sequences of semantically related words in a text, and finally all content words from a given document. The combination of all these features represents the global context.

Similarly, local contexts are generated by using semantic categories. However, the sentence context has some different features: grammatical dependency relations are produced by using standard NLP tools. And concreteness ratings [10] for all content words are included. As well as the semantic relatedness, represented by the cosine similarity of their LDA topic distributions.

Results of the experiment show that the global contexts do not provide significant improvement over the local contexts. Thus, focusing on improvements in creating the local context might be a promising procedure for detection. However, Jang et al. suggest that, with some adjustments in the original method, the global context could also yield better results.

3.4 Selectional Preferences

Methods based on selectional preferences work by relating how semantically compatible predicates occur with particular arguments. For instance, the verb *ride* usually prefers *horse* or bicycle as an object, instead of *rainbow* or *lightning*. Finding these violations has been used in several publications to find metaphors. This section will show some approaches of how to detect violations of selectional preferences.

Two Approaches to Metaphor Detection [22]

MacWhinney et al. examine the application of two approaches to metaphor detection, specifically in the target domain of *economic inequality*. One approach is based on the Common Semantic Features method, which will be discussed below, and the second is a corpus based analysis. Since the CSF procedure will be described in a later review, this one will focus on the corpus based approach.

In their publication, MacWhinney et al. hypothesize that it is possible to use tools like SketchEngine [9] to locate highly frequent metaphors for a desired target domain, given a large

[6] FrameNet is a lexical database for the English language created by the International Computer Science Institute in Berkeley.

[7] Topic modeling is a natural language processing technique that is able to learn, recognize, and extract topics across a collection of documents. For further details on this see: Rehurek, Radim, and Petr Sojka. "Software framework for topic modelling with large corpora." In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. 2010.

[8] Lexical chains describe methods used to link significant words that are about the same topic.

[9] SketchEngine is a text analysis tool for large corpora. It can be used to extract different data from text, such as collocation, concordances, keywords and so on. `https://www.sketchengine.eu/`

enough corpus. For this, they extract highly frequent grammatical relations of a given target keyword such as *poverty*. These relations may include phrases like: object-of (combat poverty), subject-of (poverty cripples), adj-subject-of (staggering poverty), pp-obj-by (enslaved by poverty). However, this is where the computational extraction stops and a human analyst has to decide whether collocations are metaphorical or not.

The corpus based extraction has a promising premise, but is limited to being only semi-automatic. However, it might be used as a basis for other procedures, providing possible metaphor candidates that can be further analysed by an automatic system.

Language-independent ensemble approaches to metaphor identification [14]

Dunn et al. developed a modular identification system that runs multiple algorithms in parallel and combines their results. The basis, however, is the previously described selection preference violation method. At its core, the system utilizes a set of classifiers to identify metaphors by looking at the amount of overlap between source and target words in terms of category information from a corpus.

Each of these so-called *Category Profile Overlap (CPO)* classifiers uses the same set of resources, but differs in the parameter settings for the algorithm. For each language, the algorithm checks a corpus for concrete associated nouns, then determines the most common semantic categories for those nouns, and finally compares the overlap of these categories between the source and target domain, identifying a metaphor when the level of overlap is low.

A separate system is used to combine the votes. It classifies metaphor candidates by looking at the results of several input CPOs. When performing classification, it returns the value of the majority. One major advantage of a modular system is, that new identification methods can be included easily. However, the language-independence - as claimed by the authors - might not hold up, since many other approaches use very specific resources only available in certain languages.

3.5 Machine Learning

Since Machine Learning implies some sort of classification, these methods work by finding appropriate features to train classifiers to identify metaphors. For further details on the topic, see the Machine Learning and Neural Networks section.

Cross-lingual metaphor detection using common semantic features [35] and *Metaphor detection with cross-lingual model transfer* [34]

This approach by Tsvetkov et al. establishes the concept of *Common Semantic Features* for metaphor detection. It describes a set of features used to train a logistic regression classifier. Other than the already mentioned approaches, it does not depend on *a priori* defined target domains and relies only on the word-level features to classify sentences. In the experiment, the method is also applied on non-English text, using only a dependency parser and a target English dictionary. The features for the classifier are: semantic categories of a word, extracted

from WordNet; the degree of abstractness of a word, obtained from the MRCPD database [36]; and any types of named entities, if present. All of these features are used to train a metaphor classifier using labeled English subject–verb–object relations.

Not relying on a predefined target domain is the major advantage of this method, compared to the previously seen approaches, making it more adaptable and resilient to biased input data. It has also been shown to work well in a cross lingual experiment, without extensive resources for a given language. In the 2013 publication the authors only focus on sentences in which verbs are used metaphorically, extensions on the method were made in the 2014 publication.

Different texts, same metaphors: Unigrams and beyond [17]

Klebanov et al. use a supervised machine learning system to classify content words in a text as either being used metaphorically or not. For the training and testing of the classifier, the previously described VU Amsterdam Metaphor Corpus and two sets of essays written by college graduates, have been used. The features included in the model were: unigrams without stemming or lemmatization, POS tags, concreteness ratings [10] and topics derived from a 100-topic Latent Dirichlet Allocation model.

By experimenting with different features for different training data, Klebanov et al. show that unigram features have a strong impact in the model's performance, while POS features help improve recall across all datasets. By using the freely available VUAMC for evaluation this procedure is more reproducible, compared to custom made evaluations. This work was then extended in 2016 [18] to investigate the effectiveness of semantic generalizations for metaphoricity classification of verbs, in which different verb classes and roles were added as features. This 2016 publication was also used as a baseline for NAACL 2018 Metaphor Detection Shared Task. See NAACL 2018 section.

Token-level metaphor detection using neural networks [12]

Dinh et al. use neural networks in combination with word embeddings for a token level classification. The previously described VU Amsterdam Metaphor Corpus has also been taken as main basis for the training and testing, however, only the most clear cut annotation of "metaphor-related word" has been used, labeling everything else as literal tokens. Furthermore, they have focused on a range of content tokens such as nouns, verbs, adjectives and adverbs. Features for training the classifier are therefore: a concatenation of embeddings surrounding each content token (with a window span of 5), concreteness ratings [10], as well as POS tags.

Dinh et al. confirm that machine learning approaches to metaphor detection show promising results without any additional features other than basic tokens, in this case word embeddings. They suggest, that the next logical step consists of experimenting with more advanced network structures such as Recurrent Neural Networks, specifically Long Short-Term Memory networks. All of these concepts will be discussed in the Machine Learning and Neural Networks section. This publication will be one of the main reference points for the practical implementation presented in this work.

The following section is a collection of computational resources used in metaphor detection,

many of which have already been briefly mentioned.

4 Computational resources for metaphor identification

This section will give an overview of computational resources for working with metaphors, with the main focus on English resources, since they are more abundant and usually larger in size. All of the resources listed here are both human- and machine readable, as well as available for the general public.

For details on how each of these resources is used in identification, see the Computational methods for metaphor identification section.

4.1 WordNet

WordNet is a lexical database for the English language, created by the Princeton University. It groups nouns, verbs, adjectives and adverbs in groups of cognitive synonyms, so-called synsets. Each of these sets express a different concept. By linking them together by means of semantic relations, a network of meaningfully related words (concepts) is created. As of September 2018 it contains 155.327 words organized in 175.979 synsets for a total of 207.016 word-sense pairs. The WordNet can be accessed online or downloaded via the Princeton University website, and many programming languages offer local interfaces (APIs) for an easy access. Here a shortened example of a synset:

Word: bank

- Noun: sloping land (especially the slope beside a body of water)
- Noun: depository financial institution
- Verb: deposit, bank (put into a bank account)

`https://wordnet.princeton.edu/`

4.2 FrameNet

FrameNet is a lexical database for the English language, created by the International Computer Science Institute in Berkeley. It is based on the theory of meaning called Frame Semantics, in which the meaning of words can be understood on the basis of a semantic frame: a description of a type of event, relation, or entity and the participants in it (so-called frame elements). Words that evoke a frame are called lexical units. An example of a FrameNet entry would be:

Semantic Frame: Emotion_heat

Description: This frame contains verbs that describe emotional experiences and participate in the locative alternation, as in the following examples: I was boiling with anger.

Frame elements: Emotion, Experiencer, Seat_of _emotion

Lexical units: boil, anger, ...

`https://framenet.icsi.berkeley.edu/fndrupal/`

4.3 VerbNet

VerbNet is an online verb lexicon for English, created by the University of Colorado Boulder. It is organized into verb classes. Each verb class is described by its thematic roles and frames, consisting of a syntactic description and semantic predicates with a temporal function. It also contains mappings to other lexical resources such as WordNet and FrameNet. An example for the VerNet entry of *breathe-40.1.2-1* is:

Thematic roles: Agent [+animate], Theme, Destination

Frames:

- NP V: Paul breathed. semantics body_process(E, Agent) emit(during(E), Agent, ?Theme)

`https://verbs.colorado.edu/~mpalmer/projects/verbnet.html`

4.4 SEMAFOR and open-sesame

SEMAFOR (Semantic Analysis of Frame Representations) is a tool for automatic analysis of the frame-semantic structure of English text. Based on FrameNet, it attempts to find which words in a text evoke which semantic frames, and to find and label each frame's arguments. However, it is no longer being maintained.

`http://www.cs.cmu.edu/~ark/SEMAFOR/`

`https://github.com/Noahs-ARK/semafor`

The newer version of the Frame-semantic parser is called open-SESAME. It is also based on FrameNet and uses a recurrent neural network to detect frames.

`https://github.com/Noahs-ARK/open-sesame`

4.5 TroFi Example Base

The TroFi Example Base is a dataset of literal and non-literal usage for 50 verbs, which occur in 3.737 sentences from the Wall Street Journal corpus [10]. It was built using the TroFi (Trope Finder) algorithm [9], which attempts to separate literal usages of verbs from non-literal ones. For each sentence in the dataset, the target verb is labeled either L (literal) or N (non-literal), according to the sense of the verb that is invoked by the sentence. For instance, the verb *strike* has the following entries:

[10]The Wall Street Journal corpus is compiled from the Wall Street Journal and contains 25 million words

- L: Sometimes it struck without warning
- N: His native caution strikes a responsive chord with the average German

`http://www.cs.sfu.ca/~anoop/students/jbirke/`

4.6 MRC Psycholinguistic Database

The MRC (Medical Research Council) Psycholinguistic Database contains 150.837 words with 26 different attributes, including: concreteness, familiarity, imagery, etc. These attributes have been gathered from multiple psycholinguistic experiments, and can be used to semantically enrich tokens. More specifically, concreteness reflects how concrete or abstract a word is. Imageability reflects how easy it is to construct a mental image of a word and word familiarity reflects how commonly it is experienced. For a full overview of all attributes see [36]. However it is noteworthy, that not every property is available for every word. For example, only 8.228 words are annotated for their concreteness.

`http://websites.psychology.uwa.edu.au/school/MRCDatabase/uwa_mrc.htm`

4.7 Concreteness Word Ratings

This dataset of concreteness ratings contains 37.058 English words and 2.896 two-word expressions (e.g. golden retriever). The data was obtained by a study using internet crowd sourcing. Here, concreteness represents the degree to which a word refers to a perceptible entity [10]. Other ratings include: age of acquisition and affective ratings (valence, arousal, dominance). Here an example from the data:

```
Word Bigram Conc.M Conc.SD Unknown Total Percent_known SUBTLEX Dom_Pos
roadsweeper 0 4.85 0.37 1 27 0.96 0 0
traindriver 0 4.54 0.71 3 29 0.90 0 0
tush 0 4.45 1.01 3 25 0.88 66 0
```

`http://crr.ugent.be/programs-data/word-ratings`

4.8 ProMetheus Corpus

The ProMetheus corpus, created by Özbal et al. [25], is a dataset consisting of English proverbs and their equivalents in Italian. For example: “actions speak louder than words”. It contains 1.054 proverbs, with a total of 4.978 tokens for English and 4.098 tokens for Italian. In addition to the word-level metaphor annotations for each proverb, the corpus also contains information such as the metaphoricity degree of the proverb, its meaning and the century that it was first recorded.

The metaphoricity degree was decided by independent annotators and expressed in 0 (completely literal), 1 (slightly metaphorical) and 2 (very metaphorical). The metaphor annotation was

created using the Metaphor Identification Procedure VU, described in the Theory section. The corpus is publicly available upon request to the authors.

4.9 VU Amsterdam Metaphor Corpus (VUAMC)

The VU Amsterdam Metaphor Corpus (VUAMC) contains a selection of excerpts from the BNC Baby corpus [11], that have been annotated for metaphors. It was created using the Metaphor Identification Procedure VU, described in the Theory section. As of now, it is the largest available hand-annotated corpus for all metaphorical language use, regardless of lexical field or source domain [29]. There are four text categories included (academic texts, news texts, fiction, and conversations), each comprising about 50.000 words. All categories are annotated for the following relations to metaphor: indirect metaphor, direct metaphor, implicit metaphor and borderline cases. Here an example phrase from the VUAMC:

```
<s n="173">
 <c type="PUQ">’ </c>
 <w lemma="would" type="VM0">would </w>
 <w lemma="cease" type="VVI">cease </w>
 <w lemma="to" type="TO0">to </w>
 <w lemma="haunt" type="VVI">
  <seg function="mrw" type="met" vici:morph="n">haunt</seg>
 </w>
 <w lemma="the" type="AT0">the </w>
 <w lemma="labour" type="AJ0">Labour</w>
 <w lemma="party" type="NN1">Party</w>
</s>
```

http://www.vismet.org/metcor/search/showPage.php?page=start

http://ota.ahds.ac.uk/headers/2541.xml

This corpus will serve as the main resource for the practical work presented in the following section.

5 Neural network metaphor detection

As shown in the Literature Review section, there are many different approaches for metaphor detection. This work will apply machine learning techniques to - metaphorically speaking - tackle the problem. Simply put, machine learning means: training a system to perform a certain task, without it being explicitly programmed. Such a system could, for instance, be trained to recognize part-of-speech tags by looking at large amounts of annotated text data. This is referred to as supervised learning, and it is a common practice in NLP [37]. In recent

[11] The BNC Baby is a four million word sampling of the 100 million word British National Corpus

years, machine learning has proven to be a very effective method for various classification tasks across many research fields. Listing the numerous publications from almost all sciences would be impossible, as well as mentioning all companies that work with machine learning techniques nowadays. Machine learning often sits at the intersection of engineering and science.

However, there has also been some criticism [11]. Any machine learning approach usually requires large datasets to train the classifier, this is problematic in domains where such data is sparse or does not exist at all. As it is in the domain of metaphor detection, for instance. Which leads to a second limitation: trained classifiers are mostly highly specialized to perform well on one task, which can lead to them only performing well in a controlled setup. When applied in a productive environment, the classification results might vary. The largest concern, however, is with a major lack of transparency. That means, compared to an algorithm, i.e. a formal set of instructions, machine learning often times abstracts the problem at hand away. A set of features and parameters is defined beforehand on which the model is then trained, this often fails to answer the question of how the underlying mechanics of the problem work. This is why people commonly refer to machine learning as a black box.

Nevertheless, when looking at the performance of recent metaphor detection methods [13], a machine learning approach might yield more promising results than other systems. This argument is reaffirmed when studying the results of the *Workshop on Figurative Language Processing* [8], which was held in June of 2018. The workshop also included a shared task on metaphors detection, in which many of the participants successfully applied machine learning architectures.

5.1 NAACL Shared Task

The goal of the shared task conducted by the NAACL was to detect, at a word level, all content-word metaphors in a given text, including also a separate evaluation for verb-tokens only [4]. Overall, there were a total of 32 submissions by 8 unique teams. The training and evaluation was done on the previously mentioned VU Amsterdam Metaphor Corpus, for which the NAACL provided subsets of tokens that had to be correctly classified by the participants. These subsets were shared among all participants to provide a common starting point and comparable results. This shared baseline contained an identifier for each token (text id, sentence id, token id) and the corresponding metaphor label of 0 or 1 (1 indicating a prediction of metaphor). For example:

```
verb_tokens_test_predictions.csv
a1e-fragment01_12_3,1
a1e-fragment01_12_21,0
a1e-fragment01_12_33,1
```

The NAACL also provided two baseline performances to compare results against. The highest of the two baseline performances for the detection in the "all-POS (over all categories)" task was: precision: 0.521, recall: 0.657, F1-score: 0.581. The highest performance in the "Verbs

(over all categories)" task was: precision: 0.527, recall: 0.698, F1-score: 0.600 [12]. They also provided a set of features used to construct the baseline classification model for prediction of metaphor and non-metaphor classes at the word level, as well as instructions on how to replicate these baselines [5]. All results and more detailed descriptions on the participants can be found in [19].

In light of this recent event, the method described here will also be trained on the VU Amsterdam Metaphor Corpus, and then be evaluated on the NAACL shared task baselines for comparable results. The architecture will be a Recurrent, Long Short-Term Memory Neural Network trained for the task of metaphor detection. All details of the model, as well as the implementation and the data preprocessing can be found in the Implementation section. The following section will give a very brief overview of the main concepts involved in machine learning and neural networks.

5.2 Machine Learning and Neural Networks

Many NLP tasks include some form of classification. Examples are POS tagging, sentiment analysis, readability scoring, or named-entity recognition, and as mentioned before: machine learning at its core is about classification. These techniques give computers the ability to learn from input data and produce output classifications for new data. Machine learning approaches can be classified into three broad categories, which are: supervised learning, in which annotated data (e.g words and their corresponding sentiments) is used to train a classifier to identify new unseen data; unsupervised learning, in which the classifier tries to find structure in data on its own, thus discovering hidden patterns; and reinforcement learning, which is more goal-oriented and gives the system rewards or penalties while it is trying to solve a task, e.g. playing chess. All of these have been used in various NLP tasks. To quote Young et al. [37]:

> [..] a simple deep learning framework outperforms most state-of-the-art approaches in several NLP tasks such as named-entity recognition (NER), semantic role labeling (SRL), and POS tagging. Since then, numerous complex deep learning based algorithms have been proposed to solve difficult NLP tasks.

Neural Networks

One group of algorithms used for machine learning is called *Artificial Neural Networks*. These algorithms model the data by using graphs of artificial neurons, those neurons are a mathematical model that mimics approximately how a neuron in the brain works. Artificial neural networks have generated a lot of excitement in research and industry, thanks to many breakthrough results in speech recognition, computer vision and text processing. At its core, a neuron takes one or more inputs and produces a single output. Each input is separately weighted, and the sum is passed through a function known as an activation function. This is to simulate the biological neuron which can either be active or not, see Figure 1. During the training phase, the correct

[12] Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall is the ratio of correctly predicted positive observations to the all observations. The F1 score is the weighted average of Precision and Recall

value for each input weight is learned by a concept known as gradient descent. The gradient descent algorithm tries to find the minimum of the models loss function, meaning the measurement of how good the model performs. During many trials, the model adjusts each neuron's weights and receives feedback from the loss function, gradually improving its performance.

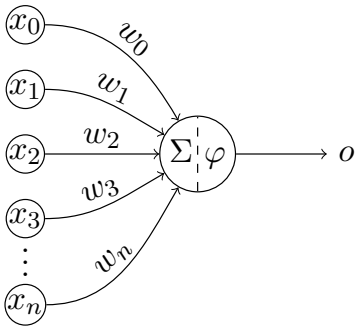

Figure 1: Single Neuron

On its own a single neuron cannot solve complex tasks. However, when put together, many neurons can recognize patterns in data and produce desirable classifications. These networks are created by interconnecting multiple neurons in different layers [13]. Complex networks like this can learn more features from a given input and produce more than one binary output (compared to a single neuron). See Figure 2 for a more complex neural network.

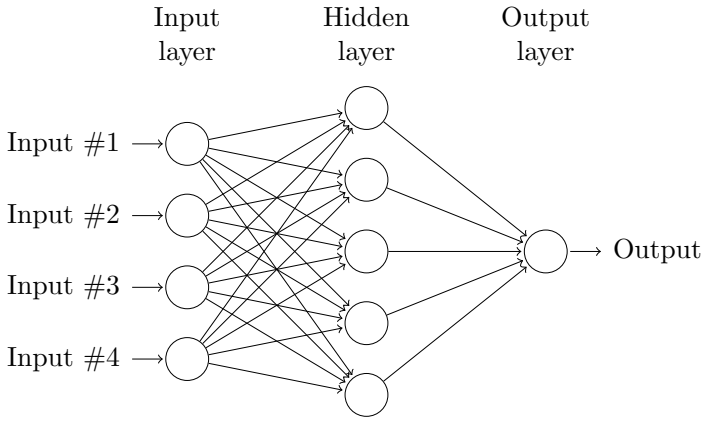

Figure 2: Multi-Layer Neural Network (feedforward)

In the task of detecting metaphors, the input layer may represent a token (as a mathematical vector) and the output will be a numerical classification (metaphor or non-metaphor). An input vector can then be enriched with additional features by concatenating numerical values (e.g. category of a POS tag). This enables the network to take more contextual information into account. When working with neural networks in NLP, the input tokens are usually transformed into word embeddings, to create an efficient semantic representation.

Recurrent Neural Networks (RNN)

As there are different tasks to be solved, there are different forms of neural networks tailored to a certain type of problem. One of those networks is called a *Recurrent Neural Network*

[13] When a neural network has one or more hidden layers it is called a Deep Neural Network

(RNN), and it has been successfully applied to many NLP tasks due to its ability to work well with sequential data. Examples would be: time series data or a stream of data which is interdependent, such as combinations of lexical items. An RNN works on the principle of saving the output of a layer and feeding this back to the input to help in predicting the outcome of the layer. This enables the network to store information from (theoretically) arbitrary long time (or input) ago, which again means more contextual information. In practice, however, these simple RNN networks suffer from the infamous vanishing gradient problem [14], which makes it hard to learn and tune the parameters of the earlier layers in the network [37].

Long Short-Term Memory (LSTM)

To overcome these issues in the RNN architecture, the concept of *Long Short-Term Memory (LSTM) units* was introduced. In simple terms, these units extend the recurrent network with additional networks (often called gates, cells, or gated cells), which act as a memory. A gated cell makes decisions about what to store, and when to allow reads, writes and removals. Those gates act on the signals they receive, and similar to the neural network's nodes, they block or pass on information based on its strength and importance, which they filter with their own sets of weights. Those weights, like the weights that modulate input and hidden states, are adjusted via the recurrent networks learning process. An RNN composed of LSTM units is often called an *LSTM network* or *Recurrent LSTM network*. As mentioned before, this architecture has been successfully applied to many tasks involving sequential data.

Token-level metaphor detection can be treated as a sequential tagging problem, like POS tagging or named-entity recognition. Thus, using a neural network architecture that is specialized in working with sequential data might yield better results, when compared to approaches with hand-crafted target domains. The neural network architecture might also be more adaptable when receiving training data tailored to specific metaphor source or target domain. A machine learning approach also has the advantage of gradually introducing new features for the input layers, such as the previously described concreteness ratings in the Resources section.

In the next section details on the implementation will be provided.

5.3 Implementation

This section includes details on the implementation and design decisions made. The entire implementation was done in Python using the open-source neural network library Keras [2], with the math library TensorFlow as backend. TensorFlow is an open-source machine learning library, it is used for both research and production at Google. The source code for this thesis, as well as an interactive Jupyter Notebook containing details on the implementation, can be found on GitHub [7].

As described before, the VUAMC is used as dataset for training and evaluation. The full XML file could not be included into the repository due to licence restrictions. However, it is

[14]The vanishing gradient problem occurs, when the gradient of the loss in large enough neural network is calculated via backpropagation. When calculating the final gradient in the backpropagation, it can explode or vanish due to repeated multiplications of results.

freely available online and functions to download and preprocess the data are included in the repository. Please refer to the Readme file provided in the repository for details on this. A script for converting the full XML file into the shared subset was provided by the NAACL. The final datasets include 12.122 sentences for training and 4.080 sentences for testing, both in the CSV format. These sentences were selected by the NAACL, and both sets come with unique identifiers for each sentences. The NAACL also provided two sets of labels for the corpus, one including only verb tokens and one all-POS tokens. These baseline files also contained an identifier for each token (text-id, sentence-id, token-id) and the corresponding metaphor label of 0 or 1 (1 indicating a prediction of metaphor). For example:

```
verb_tokens_test_predictions.csv
a1e-fragment01_12_3,1
a1e-fragment01_12_21,0
a1e-fragment01_12_33,1
```

To better manage the datasets during runtime, a "VUAMC" class was created. This class contains functionality to load all CSV data (for both test and training) into Python dictionaries, to leverage the provided identifiers. It also provides simple functions for extracting all tokens and corresponding labels. Further features included POS tags for all sentences, using the OntoNotes 5 version of the Penn Treebank tagset. The tags were generated using the Python library *spaCy* and are included in the repository as CSV files, which saves time when loading the data by skipping a repeated tagging process. Functions to recreate the POS tags are included as well.

Once instantiated, VUAMC objects are then used to enrich and normalize the data for the neural network, since input sequences for neural networks require a fixed length. Thus, all sentences are limited to a pre-defined length, by either repeatedly breaking up longer sentences or by adding padding values to shorter sentences. The average sentence length in the training data is 20 tokens, with a variance of 192.96 and a standard deviation of 13.89. To capture these deviations, the maximum sentence length is set to 50. Shorter sequences receive a right-sided padding, with "__PADDING__" as special string for identification and corresponding labels are padded using "-1" as value. Since these values never occur in the dataset, they can be easily masked during the training phase.

After the padding is applied to normalize the sequences, all tokens are encoded with word embeddings to provide a dense semantic vector representation. The special padding strings receive a zero vector with the corresponding embedding dimension. For this, an abstract class "Embeddings" was implemented, which allows for the creation of polymorph subclasses for different embedding types. During the training phase different pre-trained embeddings can be used to analyse differences in performance. In this work the following word embeddings were used: Facebook's FastText vectors, trained on Wikipedia with 300 dimensions; and Google's word2vec vectors, with also 300 dimensions. The vectors are loaded using the *pymagnitude* library, which allows for lazy loading to ensure optimal memory usage and provides functionality to find most similar vectors for out-of-vocabulary tokens. Due to the large file sizes, the vectors could not be included in the repository. See the Readme file for details on obtaining the vectors.

The architecture of the neural network is a bidirectional RNN LSTM with 100 dimensions, with a 0.25 recurrent dropout rate [15]. Bidirectional RNNs, extend unidirectional RNNs by introducing a layer, where the directed cycles enable the input to flow in opposite sequential order. While processing text, this means that for any given word the network not only considers the text leading up to the word but also the text thereafter. Other systems in the shared task using bidirectional LSTMs were: Bizzoni et al., Pramanick et al., Mykowiecka et al. and Stemle et al. [19]. The optimizer used is RMSProp, which is able to work with mini-batches (subset of n training examples) leading to a more efficient gradient computation. And it has been shown to produce a consistent performance [26]. The output is a dense layer with a softmax activation function, which squashes the outputs of each unit to be between 0 and 1, to predict the binary metaphor label sequences. See Figure 3 for the final network architecture.

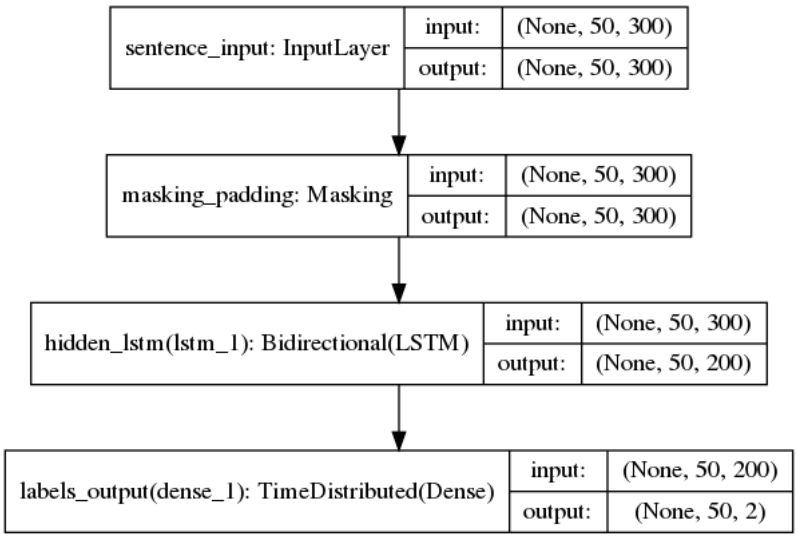

Figure 3: Neural Network Architecture

The loss function is a weighted categorical crossentropy, assigning a larger loss weight to the positive class (i.e. the metaphor class). This is done because the dataset contains a much higher frequency of non-metaphorical tokens than metaphor tokens, since each sentence includes only one or two tokens in this category. Without this adaptation, the model will almost exclusively choose a non-metaphor token, because the final evaluation will still yield a very good result when only predicting the higher frequency class. A look at the distribution of classes in the training corpus shows this imbalance clearly, see Table 1.

Non-metaphor tokens (Verbs)	152.829	96.96%
Metaphor tokens (Verbs)	4.793	3.04%
Non-metaphor tokens (All POS)	166.983	93.80%
Metaphor tokens (All POS)	11.044	6.20%

Table 1: Class Distribution - Verbs

The weight for each class is based on the frequencies of all classes, using the ratio of class-

[15] Dropout means randomly ignoring neurons during the training phase, which is making the network more robust when encountering unseen data. Recurrent dropout randomly drops the connections between the recurrent neurons.

frequency to majority class-frequency and to handle extremely uneven weights a smoothing factor can be applied during the calculation. The training then was done twice with a smoothing factor of 0.0 and 0.1. The algorithm for class weight calculation is:

```
  counter = Counter(classes)

  if smooth_factor > 0:
    smoothing = max(counter.values()) * smooth_factor
    for key in counter.keys():
      counter[key] += smoothing

  majority = max(counter.values())

return {cls: float(majority / count) for cls, count in counter.items()}
```

To further combat any bias and overfitting during the training phase, a k-fold cross-validation was applied to the dataset, in which the original sample is randomly partitioned into k equal sized subsamples, while also shuffling the entire set and repeating the training each time. For reproducibility the shuffling was done with a fixed seed. Metrics to measure the performance of the model are: precision, recall, F1-score and accuracy.

After training the model, the predictions are made on the separate test sentences. For this, all sentences are again normalized and converted to lists of word embeddings similar to the training sentences. The model will then predict the token labels as real numbers between 0 and 1, which are then converted into binary labels. To compare the results with the gold standard provided, the predictions for each token are written into a CSV file, similar to the files provided by the NAACL. Metrics calculated from comparing these two files are: precision, recall and F1-score. The next sections will present the results.

6 Results

The results were generated with the previously described bidirectional RNN LSTM model, and different word embeddings were used for comparison. For validation and control purposes a set of randomly generated embeddings was also included. The system was separately trained once with POS tags, to test the impact of word classes as features, and once with a class weight smoothing of 0.1, to examine the impact on the performance [16]. This was done once for the verb tokens and once for all-POS tokens.

The settings for the training phase of the neural network were as follows: batch size of 32, amount of epochs 5 with an 8-fold split for the training data, and a recurrent dropout of 0.25. For further details on the neural network and settings, see the interactive Jupyter Notebook which was used to generate the results [7]. All results, as well as the baselines from the shared task, are shown in Table 2.

[16]The training was done on a TUXEDO InfinityBook Pro Intel Core i7-8550U CPU 1.80GHz with 8GB RAM

Model	Verbs Testing			All POS Testing		
	P	R	F	P	R	F
Baseline 2	.527	.698	.600	.510	.696	.589
Baseline 1	.510	.654	.573	.521	.657	.581
Random	.296	.123	.174	.183	.473	.263
FastText	.495	.733	.591	.398	.841	.541
Word2Vec	.412	.705	.520	.388	.823	.527
Random + Smoothing	.271	.118	.164	.188	.234	.208
FastText + Smoothing	.489	.799	.607	.493	.706	.580
Word2Vec + Smoothing	.437	.731	.546	.486	.694	.571
Random + POS	.183	.432	.258	.183	.432	.258
FastText + POS	.489	.799	.607	.438	.785	.562
Word2Vec + POS	.453	.732	.559	.417	.753	.536

Table 2: Performance of different features

The best overall results were achieved when using the FastText embeddings and a 0.1 smoothing for the weighted loss. While POS tags as features still improve the results, when compared to the no-POS baseline, the overall improvement is still small. This may be due to a portion of tokens from conversation texts, which are often colloquial. The sentences in these transcripts are sometimes missing words, have non-grammatical structure, or are wrongly split. Examples are: "Or on him." (bpa-fragment14) or "Seven thirty for eight" (c8t-fragment01).

The Word2Vec embeddings yielded a similar, but consistently lower, performance. This was expected since the FastText embeddings have been shown to produce better results in different tasks [23]. As also expected, the random vectors did not perform well and thus validate the pre-trained word embeddings used. The baseline of the NAACL Shared Tasked could, however, not be reached, which is most likely due to the class imbalance not being properly mitigated against. The results are still a promising start for further extensions.

Several strategies might be used to combat the imbalance for further experiments. When staying within the conditions of the shared task, oversampling might be an effective option. Oversampling can be used to alter the class distribution of the training data, and it is used to deal with class imbalance by reusing data points. The main disadvantage with oversampling is that by making copies of existing examples, it may lead to overfitting. This could lead to a better score in the task, but not necessarily a better performance on real data.

Another solution might also consist in extending the neural network with Conditional Random Fields (CRF) [33]. CRF are a discriminative model, and their underlying principle is that they apply logistic regression on sequential inputs. They can be used on top of the recurrent neural network to better predict the sequential labels. A further improvement might be the inclusion of more features, to better capture the essence of metaphors, as described in the theoretical introduction. These might be the imageability and or concreteness ratings for certain or all tokens in the training data.

Outside of the conditions of the shared task, the training data could be extended by using the entire VUAM Corpus, or a more balanced subset. This may also include different sampling

methods or even adding more data from different datasets, for example the previously mentioned ProMetheus corpus. Using not only a binary classification, but all different types of metaphors in the VUAM Corpus might also change the distribution of classes. This, however, needs to be further analysed.

Nonetheless, the current model provides a solid basis for further experiments, that can easily be improved upon.

7 Summary and Conclusion

In this work, an approach for supervised metaphor detection was presented. This was done by combining pre-trained word embeddings with a neural network architecture. The linguistical foundation for this was Lakoff and Johnson's publication *Metaphors we live by*. It was shown how this theoretical framework can be operationalized for corpus creation and to implement computational methods for metaphor identification. Together with this theoretical foundation, a general overview of practical NLP methods has been given, as well as an extensive list of resources for the task. The main focus was put on the Metaphor Identification Method by the Pragglejaz Group and the resulting VU Amsterdam Metaphor Corpus.

Furthermore, the conceptual foundations of machine learning in natural language processing have been demonstrated. All details of the resulting neural network implementation are openly available on GitHub. Results of the model are solid, yet still improvable, and various strategies for further improvements have been given. As in many neural network architectures, an important step will be to isolate features that contribute to the model's performance.

Generally speaking, future effort should also be put into making metaphor detection methods more accessible outside of the experiments in which they were used in. Many of the implementations presented here are highly specialized and sometimes require a complicated setup, making them impractical for a productive adoption. One goal can - and should - be to make them a valuable addition to the hermeneutic toolbox of discourse analysts. When humans can be aided by computational systems in analysing non-literal language, it might bootstrap the creation of further resources in the field of metaphor research. This, in return, can lead to the development of better tools. For this, the practical part of this work was used in the project "Demystifying Multilingualism" [1], in which the academic discourse on multilingualism was analysed. Details on the productive implementation used for the project are outside of the scope of this work however.

In conclusion, non-literal language remains an exciting research topic. The ever growing linguistic research community, as well as the commercial interest in human-machine communication will surely continue the development of smarter language systems. As the collection of publicly available resources grows and advances in machine learning are made by a large open-source community, the next years will likely see a breakthrough in the use and understanding of language (literal and non-literal) by computer systems.

References

[1] Demystifying multilingualism. http://portal.volkswagenstiftung.de/search/projectDetails.do?ref=93182.

[2] Keras. https://keras.io/.

[3] Metanet. https://metanet.icsi.berkeley.edu/metanet/.

[4] Metaphor shared task. https://competitions.codalab.org/competitions/17805.

[5] Metaphor shared task github. https://github.com/EducationalTestingService/metaphor/tree/master/NAACL-FLP-shared-task.

[6] spacy. https://spacy.io/.

[7] Thesis source code. https://github.com/martialblog/bachelor-thesis-code.

[8] Workshop on figurative language processing. https://sites.google.com/site/figlangworkshop/.

[9] Birke, J., and Sarkar, A. A clustering approach for nearly unsupervised recognition of nonliteral language. In *11th Conference of the European Chapter of the Association for Computational Linguistics* (2006).

[10] Brysbaert, M., Warriner, A. B., and Kuperman, V. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods 46*, 3 (2014), 904–911.

[11] Burrell, J. How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society 3*, 1 (2016), 2053951715622512.

[12] Do Dinh, E.-L., and Gurevych, I. Token-level metaphor detection using neural networks. In *Proceedings of the Fourth Workshop on Metaphor in NLP* (2016), pp. 28–33.

[13] Dunn, J. What metaphor identification systems can tell us about metaphor-in-language. In *Proceedings of the First Workshop on Metaphor in NLP* (2013), pp. 1–10.

[14] Dunn, J., De Heredia, J. B., Burke, M., Gandy, L., Kanareykin, S., Kapah, O., Taylor, M., Hines, D., Frieder, O., Grossman, D., et al. Language-independent ensemble approaches to metaphor identification. In *28th AAAI Conference on Artificial Intelligence, AAAI 2014* (2014), AI Access Foundation.

[15] Group, P. Mip: A method for identifying metaphorically used words in discourse. *Metaphor and symbol 22*, 1 (2007), 1–39.

[16] Jang, H., Moon, S., Jo, Y., and Rose, C. Metaphor detection in discourse. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (2015), pp. 384–392.

[17] Klebanov, B. B., Leong, B., Heilman, M., and Flor, M. Different texts, same metaphors: Unigrams and beyond. In *Proceedings of the Second Workshop on Metaphor in NLP* (2014), pp. 11–17.

[18] Klebanov, B. B., Leong, C. W., Gutierrez, E. D., Shutova, E., and Flor, M. Semantic classifications for detection of verb metaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2016), vol. 2, pp. 101–106.

[19] Klebanov, B. B., Shutova, E., Lichtenstein, P., Muresan, S., and Wee, C. Proceedings of the workshop on figurative language processing. In *Proceedings of the Workshop on Figurative Language Processing* (2018).

[20] Lakoff, G. The contemporary theory of metaphor. *Metaphor and thought* (1993), 202–251.

[21] Lakoff, G., and Johnson, M. *Metaphors we live by*. University of Chicago press, 2008.

[22] MacWhinney, B., and Fromm, D. Two approaches to metaphor detection. In *LREC* (2014), pp. 2501–2506.

[23] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. Advances in pre-training distributed word representations. *arXiv preprint arXiv:1712.09405* (2017).

[24] Mohler, M., Bracewell, D., Tomlinson, M., and Hinote, D. Semantic signatures for example-based linguistic metaphor detection. In *Proceedings of the First Workshop on Metaphor in NLP* (2013), pp. 27–35.

[25] Özbal, G., and Strapparava, C. A corpus of proverbs annotated with metaphors.

[26] Ruder, S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).

[27] Sardinha, T. B. Collocation lists as instruments for metaphor detection in corpora. *DELTA: Documentação de Estudos em Lingüística Teórica e Aplicada 22*, 2 (2006), 249–274.

[28] Schulder, M., and Hovy, E. Metaphor detection through term relevance. In *Proceedings of the Second Workshop on Metaphor in NLP* (2014), pp. 18–26.

[29] Steen, G. *A method for linguistic metaphor identification: From MIP to MIPVU*, vol. 14. John Benjamins Publishing, 2010.

[30] Steen, G. The language of knowledge management: A linguistic approach to metaphor analysis. *Systems Research and Behavioral Science 28*, 2 (2011), 181–188.

[31] Steen, G. J. *Finding metaphor in grammar and usage: A methodological analysis of theory and research*, vol. 10. John Benjamins Publishing, 2007.

[32] Strzalkowski, T., Broadwell, G. A., Taylor, S., Feldman, L., Shaikh, S., Liu, T., Yamrom, B., Cho, K., Boz, U., Cases, I., et al. Robust extraction of metaphor

from novel data. In *Proceedings of the First Workshop on Metaphor in NLP* (2013), pp. 67–76.

[33] Sutton, C., McCallum, A., et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning 4*, 4 (2012), 267–373.

[34] Tsvetkov, Y., Boytsov, L., Gershman, A., Nyberg, E., and Dyer, C. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2014), vol. 1, pp. 248–258.

[35] Tsvetkov, Y., Mukomel, E., and Gershman, A. Cross-lingual metaphor detection using common semantic features. In *Proceedings of the First Workshop on Metaphor in NLP* (2013), pp. 45–51.

[36] Wilson, M. Mrc psycholinguistic database: Machine-usable dictionary, version 2.00. *Behavior research methods, instruments, & computers 20*, 1 (1988), 6–10.

[37] Young, T., Hazarika, D., Poria, S., and Cambria, E. Recent trends in deep learning based natural language processing. *arXiv preprint arXiv:1708.02709* (2017).