Research Article

Metaphor identification in cybersecurity texts: a lightweight linguistic approach

Kelsey Hilton[1] · Akbar Siami Namin[1] · Keith S. Jones[2]

Received: 16 August 2021 / Accepted: 4 January 2022
Published online: 28 January 2022

Abstract
The use of metaphor in cybersecurity discourse has become a topic of interest because of its ability to aid communication about abstract security concepts. In this paper, we borrow from existing metaphor identification algorithms and general theories to create a lightweight metaphor identification algorithm, which uses only one external source of knowledge. The algorithm also introduces a real time corpus builder for extracting collocates; this is, identifying words that appear together more frequently than chance. We implement several variations of the introduced algorithm and empirically evaluate the output using the TroFi dataset, a de facto evaluation dataset in metaphor research. We find first, contrary to our expectation, that adding word sense disambiguation to our metaphor identification algorithm decreases its performance. Second, we find, that our lightweight algorithms perform comparably to their existing, more complex, counterparts. Finally, we present the results of several case studies to observe the utility of the algorithm for future research in linguistic metaphor identification in text related to cybersecurity texts and threats.

Article Highlights

- It is possible to perform metaphor identification in a lightweight linguistic approach.
- The results show promising results in comparison with more complex approaches.

Keywords Cyber security · Linguistic analysis · Natural language processing · Metaphor

1 Introduction

Metaphor has been a subject of study in cognitive psychology, linguistics, and Natural Language Processing (NLP) for years. It is a topic that is truly interdisciplinary. The widely known Conceptual Metaphor Theory (CMT), laid out by Lakoff and Johnson [1], suggests that "*the essence of metaphor is understanding and experiencing one kind of thing in terms of the other*" [1]. People naturally use terms from a concrete or well understood source of knowledge (i.e., "source" domains) and apply them to abstract, or less understood areas of knowledge (i.e., "target" domains) to transfer knowledge [1, 2]. Although the mapping of knowledge from source to target domains, known as the conceptual metaphor, is complex, linguistic metaphors are common. Research shows that metaphorical language occurs, on average, in one in three sentences [3].

This work is supported by National Science Foundation (NSF) under award numbers: CNS-1347521 and SES-1564293.

✉ Akbar Siami Namin, akbar.namin@ttu.edu | [1]Department of Computer Science, Texas Tech University, Lubbock, TX 79409, USA. [2]Department of Psychology, Texas Tech University, Lubbock, TX 79409, USA.

In recent years, metaphor in cybersecurity has gained attention [4–6]. Topics of interest include: (1) how metaphors influence our understanding of abstract security concepts?, (2) whether metaphors place limitations on our understanding?, and (3) if effective, how metaphors can inspire new approaches to communicate cybersecurity risks and problems? [6]. Work has gone into identifying metaphorical language in cybersecurity [4, 6], and how it shapes our understanding of unfamiliar concepts [6]. One study demonstrates that using metaphor in user interface design increases understanding of abstract concepts and influences online behavior of novice users [5]. Cybersecurity threats are not always well understood or recognized by end users who depend on their computer hardware and software for everyday tasks. The risk to users for missed or misunderstood threats can be life altering; thus, it is important to study the effect of metaphor in cybersecurity to find ways to reduce this risk through effective communications.

Despite the large amount of research in automatic linguistic metaphor identification, the field still faces major challenges. Research has not converged on standards or conventions for some common components of identification computation, the field does not agree on a single definition of metaphor for computational analysis, and standards for reporting are lacking [7]. These issues make it difficult to compare approaches and thus rely on outcomes and results.

1.1 Motivation: communicating the impacts of cybersecurity threats with users

While existing research explores mental models [4] and metaphors in cybersecurity [6], to our best knowledge, no research has been done to automatically extract metaphors used to communicate about individual cybersecurity threats. The underlying motivation for the present research is to discover metaphors that may be useful for creating effective communications about abstract cybersecurity concepts, in hopes that increasing understanding will encourage end users' decisions that help protect them from cyber attacks.

Although the present focus lies in cybersecurity, the applicability of the present research to NLP is also considered. The facets of the metaphor identification task that this research addresses are outlined below:

- A lack of reporting standards, common definitions, and shared resources in metaphor identification research, and the observable impact of input on the outcome, makes it difficult to compare performance and draw conclusions about the effectiveness of existing metaphor identification systems.
- Words are often polysemous, i.e., they have multiple meanings, which presents a challenge for metaphor identification systems that base their decisions on frequency data or characteristics of words.
- Metaphorical language in cybersecurity is evident, but it is not yet known what is the best data source for computationally searching for metaphor (academic papers, social media, science fiction, etc.), or if there are novel yet unidentified metaphors, which could be leveraged in future risk communication work.

The present approach is twofold: The first aim is to automatically identify metaphor in text related to specific cybersecurity threats. It is hypothesized that one can identify metaphors for a cybersecurity threat by executing existing metaphor identification algorithms and substituting large static corpora with a smaller targeted corpus generated in real time from the Web. The second aim is to contribute to computational linguistics and NLP by demonstrating that lightweight implementations of metaphor identification systems perform comparably to accepted algorithms making the capability for linguistic metaphor identification more accessible.

1.2 Research contributions

The present research created a linguistic metaphor identification system integrating widely accepted and available computational approaches for identifying metaphors using the Internet as a real time knowledge source. New approaches for metaphor identification were not created. Instead, existing algorithms that are well documented and broadly accepted were adopted. These algorithms were used, alone or integrated, to achieve more accurate and reliable outcomes. The present research developed a corpus in real time that is used for computational analysis to determine how effectively metaphors are identified with knowledge sources extracted from the Web. The system developed for the present research was tested using controlled inputs and the output of the present system was compared to that of existing systems on the same controlled data. The key contributions of this research paper are as follows:

1. Design, implement, and integrate a metaphor identification system integrated with four metaphor identification algorithms, based on existing identification algorithms, and compare the performance of each algorithm, using the same input data; to address the skepticism of existing systems' performance, created by a lack of reporting standards, common definitions, and shared resources in metaphor identification research.

2. Develop a real time corpus builder to utilize as a knowledge source to programmatically create current corpora, necessary for collocate extraction, from the Web; address the needs for modern language, including cybersecurity related text, which may not exist in hand-coded corpora.
3. Empirically evaluate the impact of word sense disambiguation on automatic metaphor identification; address the problem of polysemy in metaphor identification.
4. Analyze system output on input data specific to individual cybersecurity threats scraped from the Web pages in real time; get a sense of the type and amount of metaphors available in the text from top results from a broad search of the Web.

This paper is organized as follows: Section 2 reviews the related work in the area of metaphor detection. Section 3 presents the algorithm for metaphor identification that was developed as part of this research. The variations of the proposed algorithm is presented in Section 4. The performance of the introduced algorithm in the context for general texts is presented in Section 5. Section 6 presents three case studies in the context of cybersecurity texts with the objective of assessing the performance of the proposed model in metaphor detection in cybersecurity texts. Section 7 concludes the paper and highlights the future research directions.

2 Related work

Metaphors shape the way we understand the world. Many existing metaphor identification approaches use large static corpora, which may not contain any text related to specific domain such as cybersecurity. In this section, we discuss work related to metaphor identification, word sense disambiguation, and metaphor in cybersecurity, in preparation for a discussion of the algorithm developed during the present research in Sect. 3.

2.1 Metaphor and linguistic metaphor: definitions and different types

One of the struggles that metaphor identification research faces is a lack of a common definition of metaphor [7]. In metaphor identification literature the word "metaphor" is used inconsistently to describe cross domain mappings [8], a figure of speech [9], or a mechanism for reasoning [7, 10]. Metaphor, as a function of speech, is used to map properties from one concept to another. The conceptual mapping is known as a *conceptual metaphor*.

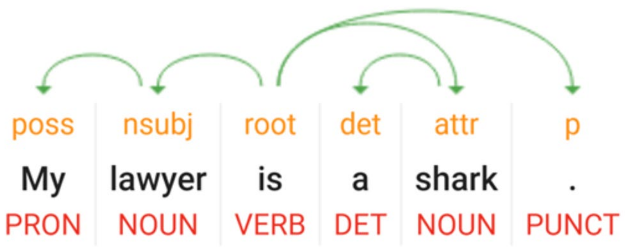

Fig. 1 Type I metaphor

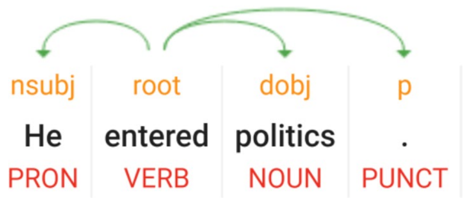

Fig. 2 Type II metaphor

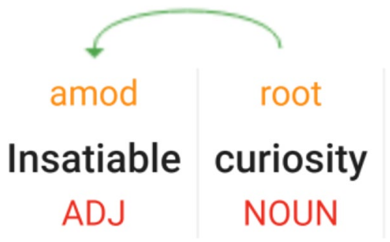

Fig. 3 Type III metaphor

An example of conceptual metaphor is "Life is a Journey." Here, different aspects of a journey can be cognitively mapped to the abstract concept of life. For example, obstacles in life are sometimes referred to as "a bump in the road" and overcoming obstacles as "everything is going smoothly" [11]. The ability to map aspects of a journey to different parts of life is what makes "Life is a Journey" a conceptual metaphor.

A linguistic metaphor is an actualized version of a conceptual metaphor in the form of a sentence, phrase, or extended piece of text [7] and can be categorized by grammatical patterns [9]. Krishnakumaran and Zhu identified three grammatical patterns typical of metaphor, which they label them as: Type I, Type II or Type III [9]. Examples of linguistic metaphor types along with an illustration of each grammatical structure are provided in Figs. 1, 2, and 3. The images of the sentence structures were generated by Google's Natural Language API demo[1].

[1] Created at https://cloud.google.com/natural-language/.

Each image represents an example sentence by labeling each word with its part of speech and dependency relations. Parts of speech are listed below each word. Grammar relations are depicted by an arrow from one word to another. The arrows represent the grammar relations and the name of each relation is provided at the end of the arrow above the word.

- **Type I:** Type I metaphors consist of a subject and an object noun with an identifying clause. This structure will always have "IS-A" relationship (e.g. "My lawyer is a shark" [12]).
- **Type II:** Type II metaphors contain an action verb together with its direct object noun (e.g. "He entered politics" [12]).
- **Type III:** Type III metaphors contain an adjective and the noun it describes (e.g. "Insatiable curiosity" [12]).

While these three syntactic patterns do not encapsulate all metaphors, they are useful to limit the corpus to sentences where metaphors are likely to appear.

2.2 Linguistic metaphor identification: state of the art

Researchers have introduced many different approaches, and some in combination, to automatically identify linguistic metaphor in text. This section briefly reviews the state-of-the-art of this line of research.

2.2.1 Selectional preference violation

Selectional preference describes the tendency for words to co-occur with other words in a specific semantic domain, or category, of related words [13]. It follows that selectional preference violation describes the case where a pair of words that are not frequently paired appear together. Selectional preference violation, as an indicator of metaphorical language, has been widely embraced [12].

Neuman et al. classifies Type I metaphors by determining whether the modified noun belongs to one of the concrete categories associated with the literal use of the adjective [13]. Krishnakumaran and Zhu use selectional preference violation based on knowledge learned from bigram frequencies on the web [9]. Haagsma and Bjerva attempt to generalize across existing selectional preference information to detect novel metaphors [14]. Mason [15] applies Resnik's selectional preference algorithm to identify preferences by domain to identify conceptual metaphors [15]. Resnik's selectional preference algorithm is represented in Algorithm 1:

Algorithm 1: Resnik's Selectional Preference Algorithm [16].

```
Input: n
Output: Sense of n
1:  Let n be a noun that stands in relationship $R$ to predicate $p$
2:  let $\{s_1, \dots, s_k\}$ be possible senses of n
3:  for $i$ from 1 to $k$ do
4:      $C_i = \{c \mid c$ is an ancestor of $s_i\}$
5:      $a_i = \max_{c \in C_i} A_R(p, c)$
6:      assign $a_i$ as the score for sense $s_i$
7:  end for
8:  order the possible senses of n, $\{s_1, \dots, s_k\}$, by their newly assigned scores, $a_i$ descending in value
9:  if the more than one sense is tied for the greatest sense score then
10:     break the tie by random choice
11: else
12:     select the sense $s_i$ with the greatest score, $a_i$, as the sense of n
13: end if
```

Despite its wide applications, there are well known issues with selectional preference violation in metaphor identification. Haagsma and Bjerva point out that it is a fundamental shortcoming that selectional preferences are based on frequency. Selectional preference data are obtained by counting how often words occur together in a certain relation [14]. Therefore, results are highly influenced by the corpus. Second, selectional preference violation is common to all types of non-literal language, not just metaphor, so the approach tends to classify all non-literal text as metaphorical. Additionally, common verbs are almost always classified as metaphorical [14, 17]. Finally, selectional preference is not helpful with pronouns, as a result pronouns increase the number of false negatives when using selectional preference violation as a metaphor identification technique [9, 14].

2.2.2 Word abstractness

The Contemporary Theory of Metaphor posits that *"metaphor allows us to understand unstructured subject matter in terms of a more concrete or at least a more highly structured subject matter"* [8]. This motivates the study of the relation between abstractness of words and metaphorical text. Turney et al. [10] hypothesized that metaphorical language is related to the abstractness of a word's context. In the Concrete-Abstract approach, Turney et al. [10] creates two algorithms, one to assign abstractness ratings to words to create a knowledge base which is used in future experiments. The algorithm used for metaphor identification utilizes feature vectors and logistic regression to determine whether a word is being used metaphorically [10]. The algorithm used to create a knowledge base of abstractness values assigns words an abstractness score between 0 and 1 by computing the sum of a word's similarity with twenty abstract words and subtracting the sum of a word's similarity with twenty concrete words. Similarity values are determined by Latent Semantic Analysis (LSA). To assign classification values of "metaphor" or "literal", Turney et al. uses a logistic regression model with feature vectors. The feature vector consists of five features:

(1) The average abstractness ratings of all nouns, excluding proper nouns;
(2) The average abstractness ratings of all proper nouns;
(3) The average abstractness ratings of all verbs, excluding the target verb;
(4) The average abstractness rating of all adjectives;
(5) The average abstractness ratings of all adverbs [10].

The logistic regression model is trained on a test dataset "to classify a word usage as literal or metaphorical" [10].

2.2.3 Combined approaches

Neuman et al. [13] created an algorithm for Type II metaphor identification, named CCO*. The pseudocode for the CCO* algorithm is included in Algorithm 2. It combines the notion of abstractness and selectional preference violation [13]. CCO* defines selectional preferences by category inclusion and classifies metaphors based on category overlap. Categories are identified for the top 100 most concrete collocated nouns with the source verb. If there are no categories that overlap with the categories of the source word, the metaphor candidate is considered metaphorical.

Algorithm 2: CCO* Algorithm [13].

```
Input: Verb, Object Noun
Output: Classification of "metaphorical" or "literal"
1: Using COCA $n$-grams, select the top 1000 collocated nouns with $v$ within a lexical window of 2
   and a Mutual Information Score (MI) of 3
2: Using Turney's Abstractness ratings, identify the 100 most concrete object nouns from the
   collection of collocated nouns
3: Using WordStat dictionary, identify the categories of the 100 nouns
4: Categorize Object Noun
5: if If none of the Object Noun categories overlap with the categories from the collocated nouns then
6:     return "metaphorical"
7: end if
8: Using ConceptNet, find the main category of Object Noun
9: if the main category is not included in the collection of categories from the collocated nouns then
10:    return "metaphorical"
11: else
12:    return "literal"
13: end if
```

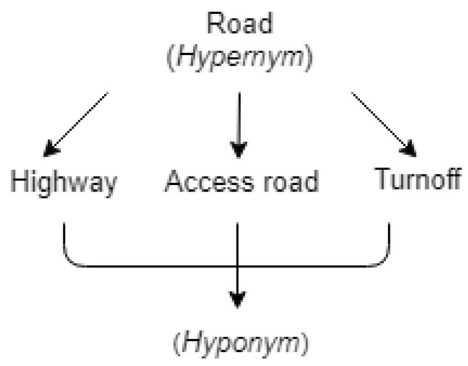

Fig. 4 Hypernym hierarchy

2.3 Word sense disambiguation

Polysemy describes a word's ability to have multiple senses. In NLP, there is a task known as word sense disambiguation, which aims to identify which sense of a word is utilized in context. A set of senses for a word can be defined by the word's dictionary definitions, a collection of synonyms from a thesaurus, or other knowledge bases, such as WordNet [18]. Word sense disambiguation is useful for machine translation, speech synthesis, and question and answer applications. Common techniques for word sense disambiguation include; dictionary-based, supervised machine learning, and semi-supervised machine learning. One of the first dictionary-based approaches to automatic word sense disambiguation was pioneered by Lesk [19]. Lesk proposed that the sense of a word can be determined by comparing the overlap of words between a target word's context and each of the target word's dictionary definition [19]. The definition with the highest number of overlapping words is the sense of the word in that instance [19]. The benefit of the dictionary approaches over machine learning techniques is that they do not require large sense-tagged corpora.

Banerjee and Pedersen created an adapted version of the Lesk Algorithm for word sense disambiguation in WordNet [20]. The Adapted Lesk Algorithm [20] utilizes the hierarchical structure of WordNet to improve on the original Lesk Algorithm [19]. Rather than just using the definition, or gloss as it is referred to in WordNet, the Adapted Lesk Algorithm also considers the glosses of related words.

Words are considered related by the Adapted Lesk Algorithm if they belong to a synset (i.e. synonym set) in the hierarchical hypernym trail. A hypernym trail is defined as the sequence of "IS-A" relations that go from specific words at the bottom to generic terms at the top [18]. The word hypernym describes the "IS-A" relationship. A hyponym describes the same relationship, but in the opposite direction; it can be thought of as a "CAN-BE-A" relationship. Figure 4 illustrates the hypernym hierarchy and the relationship between hypernym and hyponym.

2.4 Performance measurement in metaphor identification

The field of metaphor identification struggles to make progress because of a lack of shared resources, a lack of a shared understanding of the task and no clear expectations on reporting [7]. The majority of the field reports the performance of their approaches based on metrics such as precision and recall [7]. Other, less frequently reported, measures include F-score and accuracy. Dunn makes an important contribution to the field by re-implementing several algorithms to compare the impact of different features on the identification of metaphor in the same data [21]. Dunn reports, not only of F-score, but also on binary classifications; true positives, false positives, true negatives, and false negatives [21]. This is valuable because it gives insight as to why each system performs the way it does. A system may yield a high accuracy if it tends to overly classify text as metaphorical and the dataset consists of mostly metaphorical text [7], reporting on binary classifications reveals the bias [21]. Dunn also compares the systems across different genres of text [21]. What Dunn finds is that each system performs similarly, but for different reasons. The systems do not all fail and succeed on the same sentences [21]. The implication of that finding, is that the accuracy of a system is influenced by the dataset.

Gao et al. [22] present a neural model for detecting metaphorical use of words in texts. They use a bidirectional Long Short-Term Memory (LSTM) model to encode a given sentence and then a classifier based on feed-forward neural network. The LSTM part of the model predicts the metaphoricity of each word in a given sentence; whereas, the classifier predicts the binary label of each word.

Gong et al. [23] present a combined approach for detecting metaphor. The contextual representations are combined with the RoBERTa [24] model along with some other linguistic information such as WordNet. RoBERTa enables the generation of contextualized word representations to capture the context-sensitive semantics of words. The authors report F1 scores between 70.3% to 77.1% on various benchmark datasets.

Mao et al. [25] present two end-to-end metaphor identification models based on 1) Metaphor Identification Procedure (MIP) [26] and 2) Selectional Preference Violation (SPV) [27]. According to MIP, a metaphor is identified if the literal meaning of a word is different than the intended meaning; whereas, according to SPV, a metaphor is identified through a semantic difference between a target word and its context. They reported that both models outperform the state-of-the-art baselines.

Stowe et al. [28] states that even though deep learning-based models outperform the state-of-the-art of metaphor identification, these techniques suffer from complicated procedures and need large training sets. To overcome this problem, the authors propose to use syntactic features and lexical resources for providing additional high-quality training data for metaphor detection. They report that using the proposed approach improves training data and thus the classification tasks.

Stowe and Palmer [29] explore the relationship between syntactic constructions and metaphoric language. They report that integrating syntax and semantics is very essential to detect and interpret metaphor. They also report that because of diversity of contexts, it would be beneficial to use separate methods for representing metaphoric semantics with respect to the syntactic constructions involved.

Su et al. [30] present a reading comprehension paradigm for metaphor detection. The basic idea is to encode the global text context (i.e., whole sentence), local text context (i.e., sentence fragment), and questions (i.e., query word) information along with incorporating information captured through Part of Speech (PoS) features. They report competitive results obtained by their paradigm.

2.5 Metaphor generation

There exist some research work that concentrate on automatic generation of metaphor. As an example and to support linguistic analysis and metaphor detection, Klebanov et al. [31] introduce a corpus of essays written by non-native English speakers annotated for metaphor. The corpus is annotated by 240 argumentative essays that have been used as a predictor of measuring the essay quality. The corpus is the first of its own through which metaphor is used to assess the essays and their qualities.

Chakrabarty et al. [32] present Mermaid, a method to generate a metaphoric sentence by replacing relevant verbs in a literal expression. The basic idea is to transform metaphorical sentences to their literal counterpart using masked language modeling and commonsense interference. To generate high quality metaphor, a discriminator is utilized to direct and fine tune the decoding of a sequence to another one.

Gero and Chilton [33] introduce an interactive tool for generative metaphor called Metaphoria. The tool accepts concepts and then selects common poetic symbols that can be paired for the concept entered to create seed metaphor. The goal of Metaphoria is to provide unexpected but relevant ideas to assist in creative writings.

Stowe et al. [34] presents a metaphor paraphrase generation based on sequence-to-sequence mapping, called metaphor masking. In the introduced framework, the metaphoric words in the given input are replaced with metaphor masks (i.e., metaphor tokens). The framework then creates training data where the input is the masked text and the output is the original text. The evaluation based on crowdsourcing shows that the proposed metaphor is more effective in generating metaphor than lexical replacement alone.

2.6 Metaphor in cybersecurity

Metaphors are common in the cybersecurity domain and range from individual words to complex mental models. Common mental models of computer security are: 1) physical security; 2) medical; 3) criminal; 4) warfare; and the 5) market [4]. Each model contributes to the understanding of security related concepts that may otherwise be difficult to understand. Raja et al. [5] suggests that including the physical security model in security warning may facilitate the understanding of warning information, better communicate the risk and its impacts, and increase the likelihood of safe and defensive behavior [5].

2.7 The present approach

The present research aims to identify Type II metaphors related to cybersecurity threats. However, it is a challenging problem where to find text about cybersecurity threats that contains Type II metaphor. Rather than using a lot of computational resources to search for metaphors, several existing approaches were adapted to create a metaphor identification algorithm that is lightweight. The present aim is to create a methodology with comparable accuracy on a much smaller scale. In order to accomplish this, a novel approach to identifying linguistic characteristics of certain words was undertaken. Rather than using large pre-parsed corpora, a small targeted corpus was built and utilized in real time within the algorithm developed as part of the present research.

3 A computational linguistic algorithm for metaphor identification

The methodology developed as part of this research is designed to identify Type II metaphors, which are defined by a direct object grammatical relation. A hybrid approach was implemented, using existing metaphor identification algorithms. Furthermore, a real time corpus builder crawls online resources to computationally identify metaphors in text about a given context such as cybersecurity threats.

3.1 Design considerations

The underlying motivation behind the development of the present metaphor identification algorithm is to build a better user interface by automatically selecting the best metaphor for a user interface to help communicate the risk of a cybersecurity threat. This influences design decisions in the ways described below.

Level of Metaphor. The primary interest is metaphor in language rather than conceptual mappings. Therefore, the present system was designed to identify linguistic metaphors whose words can be further processed by a computer without human interpretation.

Unit of Analysis. A unit of analysis in metaphor identification is a word, relation, or sentence. A present goal was to identify text that can be transformed into a some form of representation. Therefore, the present research focused on identifying grammatical relations that contain action verbs, with the assumption that action-oriented language is more easily represented to the end users.

Syntactic Constructions. The choice of grammar relation as a unit of analysis prompted consideration of specific syntactic constructions. Krishnakumaran and Zhu [9] define specific grammatical relations typical of metaphor. The present research embraced their definition of a verbal metaphor and designed a methodology that focuses on identifying Type II metaphors.

Applicability to NLP. Shutova argues that metaphor processing systems should be designed for real-world use [7]. This means metaphor processing systems should be easy to integrate with other NLP applications, accept raw and naturally occurring text as input, work across all topics and genres, rely very little on hand-coded knowledge sources, and handle all syntactic constructions [7]. The present algorithm was designed with the goal of reducing the number of hand-coded knowledge sources and accepting raw text as input. Although the present research is centered around cybersecurity, the algorithm is not designed around the topic, making it usable across all genres.

3.2 The methodology and the algorithms

The present algorithm closely follows the CCO* approach proposed by Neuman et al. [13]. CCO* relies on four different external knowledge sources.

(1) Corpus of Contemporary American English (COCA) N-Grams corpus [35], a corpus of word sequences with their frequencies: to identify collocates;
(2) WordStat Dictionary of Semantic Categories [36], a dictionary constructed from WordNet that categorizes words included in WordNet into 44 logical groupings: to identify the categories to which a word belongs;
(3) Turney's Abstractness Ratings, the result set from Turney et al's Concrete-Abstract algorithm [10]: to rank collocations by concreteness scores;
(4) ConceptNet [37], a semantic network that represents commonsense knowledge represented by words and phrases: to identify the main sense of a word.

In the original paper, the evaluation of CCO* was limited to the study of five nouns because the frequency of data necessary for their algorithm was not available in the COCA *n*-grams corpus [13]. The present methodology only uses WordNet as a hand-coded knowledge base.

3.2.1 The methodology

Unlike existing metaphor identification systems, the present system uses the Internet to create a real time corpus to identify collocated nouns with the source verb. To overcome the limitations imposed by large corpora, the present system creates a much smaller, but targeted, corpus for each sentence that is processed (i.e., a lightweight corpus). The present system utilizes the Internet as our source of data, and the Bing API[2] to search the Internet. In summary, the present system:

(1) utilizes modifications to CCO* to define the present algorithm;
(2) reduces the number of knowledge sources from four to one,
(3) adds word sense disambiguation, and
(4) integrates a novel real time corpus builder to overcome computing resource and performance challenges imposed by corpus size.

The pseudocode for the present methodology, called CIA+, is presented in Algorithm 3.

[2] https://docs.microsoft.com/en-us/azure/cognitive-services/bing-web-search/.

Algorithm 3: CIA+ Algorithm

```
Input: Sentence, Verb, Object Noun
Output: Metaphorical classification as true or false
 1: Query Bing API for top 50 Web sites related to Verb (Algorithm 4)
 2: Extract sentences from the Web sites' content that contain Verb (Algorithm 4)
 3: Parse each sentence to find collocated nouns
 4: Use WordNet to find the synsets and hyponyms for all of the collocated nouns and the synsets of
    Object Noun (Algorithm 5)
 5: Apply the Adapted Lesk Algorithm (Sentence, Object Noun) (Algorithm 6)
 6: if the main synset does not appear in the list of synsets related to the collocated nouns then
 7:     return true
 8: else
 9:     return false
10: end if
```

3.2.2 The corpus builder

The first step in the present methodology is building a real time and targeted corpus. Building a real time corpus from the Internet is beneficial for several reasons. First, it improves overall efficiency. Large pre-parsed corpora are difficult and time consuming to create. By using the Internet, the present system can query and parse data in real time. This means that only relevant portions of text need to be parsed. Second, the wide variety of content on the Internet allows for the creation of a custom corpus, specific to the present needs. Third, by using the Internet, modern usage of the words can be easily obtained.

Next, the present system finds the nouns that frequently co-occur with the metaphor candidate's verb. The corpus builder creates a small corpus that only contains sentences that have at least one direct object relation where the relation's verb matches the metaphor candidate's verb. The present system extracts nouns from the direct object relations in each sentence and considers those nouns to be collocated with the source verb. The pseudocode for the corpus builder is presented in Algorithm 4.

Algorithm 4: Real Time Corpus Builder

```
Input: Verb
Output: List of sentences containing Verb
1: Use the query "example usages of [Verb]" as input into the Bing API and retrieve the top 50 Web
   sites.
2: The system scrapes the HTML from each of the 50 Web sites returned by Bing.
3: The HTML markup is stripped and cleaned so that only the text content is left.
4: Then, each sentence that contains verb is extracted and parsed by the Stanford Dependency parser.
5: if Verb belongs to a direct object relation then
6:     Add the sentence to the list of sentences to be output.
7: end if
```

The next step is to pre-process the sentence to identify sentences which contain the direct object grammatical relation of Type II metaphors. The present system identifies this relation by parsing the input sentence with the Stanford Dependency Parser [38] and extracting direct object relations. The Stanford Dependency Parser [38] is a program that uses the arc-standard algorithm to analyze a sentence and creates a parse tree that represents the grammatical structure of the input sentence. The parser outputs a list of Universal Dependencies [38], which represent grammatical relations in a sentence. Each relationship is a triple that relates pairs of words with a phrase label.

Consider the following example sentence:

"In a distributed denial-of-service attack (DDoS attack), the incoming traffic flooding the victim originates from many different sources[3]*".*

Given this sentence, the Stanford Dependency Parser [38] returns the Universal Dependencies and their relationship listed below where victim and flooding are having direct object relationship:

case(attack-5, In-1)
det(attack-5, a-2)
amod(attack-5, distributed-3)
amod(attack-5, denial-of-service-4)
nmod:in(originates-17, attack-5)
compound(attack-8, DDoS-7)
appos(attack-5, attack-8)
det(traffic-13, the-11)
amod(traffic-13, incoming-12)
nsubj(originates-17, traffic-13)
acl(traffic-13, flooding-14)
det(victim-16, the-15)
dobj(flooding-14, victim-16)
root(ROOT-0, originates-17)
case(sources-21, from-18)
amod(sources-21, many-19)
amod(sources-21, different-20)
nmod:from(originates-17, sources-21
...

Table 1 Algorithm comparison

	CCO*	CIA+	CIA*	CIAC	CON
Based on CCO*	x	x	x	x	
Type II metaphor	x	x	x	x	x
Real time Web Corpora		x	x	x	
Adapted Lesk Algorithm		x			
Concreteness Scores	x			x	x
Utilizes WordNet		x	x	x	

A list of Universal Dependencies with phrase labels and definitions are found in the original Stanford typed dependencies manual. The present systems' metaphor identification algorithm focuses on Type II metaphors defined by verb noun relations, otherwise known as direct object relations. The direct object relation is one of the Universal Dependency relations defined by the Stanford Dependency Parser [38] and is represented by the phrase label: **dobj**.

3.2.3 Creating wordNet synsets

The procedure used for building an ad-hoc semantic class is shown in Algorithm 5. After a set of collocated nouns is identified, an ad-hoc semantic class that consists of WordNet synsets is created. The semantic class includes synonyms and hyponyms. The latter are included in response to the idea of metaphors as *"class inclusion statements"* as proposed by Glucksberg and Keysar [39].

Algorithm 5: Create Semantic Class of Synsets

```
Input: List of Collocated Nouns
Output: Collection of WordNet Synsets
1:  Initialize empty Collection of WordNet Synsets
2:  for each Noun in List of Collocated Nouns do
3:      Retrieve WordNet $synsets$
4:      for each $synset$ do
5:          Add the $synset$ to Collection of WordNet Synsets
6:          Retrieve WordNet $hyponyms$ for $synset$
7:          Add the $hyponyms$ to Collection of WordNet Synsets
8:      end for each
9:  end for each
10: return Collection of WordNet Synsets
```

[3] Retrieved from https://en.wikipedia/org/wiki/denial-of-service_attack.

3.2.4 Filtering synsets using adapted lesk algorithm

Next, the present system uses word sense disambiguation to narrow the target noun's meaning to one synset. It uses the Adapted Lesk Algorithm [20] for word sense disambiguation. The input to the Adapted Lesk algorithm is the sentence being evaluated and the target noun. The output is a single WordNet synset that represents the sense of the word based on its use in the sentence. The pseudocode for the Adapted Lesk Algorithm is presented in Algorithm 6. The algorithm's decision is based on the ad-hoc semantic class and the target noun's disambiguated synset. If the target noun's synset is in the list of synsets that make up the class, the metaphor candidate is considered "literal". Otherwise, the phrase is considered "metaphorical".

Algorithm 6: Adapted Lesk Algorithm

```
Input: Sentence, Noun
Output: The synset that best fits Noun's use in Sentence
 1: Let lemma = lemmatized noun
 2: if lemma is not in WordNet then
 3:     return
 4: end if
 5: Let ss be a dictionary of (synset, signature)
 6: Let word_synsets be a list of synsets
 7: for each synsets do
 8:     ss += signature words of the synset
 9:     ss += signature words of the synset's hypernyms
10:     ss += signature words of the synset's hyponyms
11:     ss += signature words of the synset's holonyms
12:     ss += signature words of the synset's meronyms
13: end for each
14: Let overlap_words be a dictionary of (overlap.length, synset)
15: for each item do
16:     Let overlap_words = words in the sentence also in the set of signature words
17:     Add (overlap.length, item.synset) to overlap_words
18: end for each
19: Sort overlap_words from most to least overlap
20: return synset with the most overlap
```

4 Algorithm variations and models

This section outlines a system designed around a metaphor identification algorithm implemented in a software application. The present research's main methodology along with the developed algorithms are detailed in the previous section. In this paper, this algorithm is referred to as the Class Inclusion Algorithm (CIA*) with Word Sense Disambiguation (CIA+). In this section, two variations of CIA+ (CIA and CIAC) are presented and a model that only considers the concreteness values of the target noun (CON) are presented.

The first variation, referred to here as CIA, follows the same procedure as CIA+, but excludes word sense disambiguation. The purpose of CIA+ was to evaluate the impact of word sense disambiguation on metaphor identification. The second variation utilizes concreteness scores in conjunction with CIA*, and is referred to here as CIAC. Finally, an approach that only considers the concreteness values of target nouns, which is referred to here as CON, is presented. Table 1 outlines the differences between each approach.

4.1 Class inclusion algorithm (CIA*)

The CIA+ algorithm introduced in the previous section was developed with word sense disambiguation to address polysemy. In an effort to evaluate the impact of word sense disambiguation on the performance of the algorithm, a class inclusion algorithm without word sense disambiguation (CIA*) was created. This algorithm is identical to CIA+, except that it excludes the Adapted Lesk Algorithm. Instead of finding a "single" synset for the target noun and basing the decision about literalness based on whether the synset exists in the list of synset that belong to the collocated nouns, CIA considers "all" of

Table 2 Binary classifications and results: full data set

	True	False	True	False	No	Results			
	Pos.	Pos.	Neg.	Neg.	Decision	Acc.	Pre.	Rec.	F-score
CIA+	36	41	16	8	6	0.48	0.46	0.81	0.58
CIA*	34	33	24	10	6	0.57	0.51	0.77	0.61
CIAC	29	23	31	15	9	0.61	0.56	0.67	0.61
CON	41	46	15	3	2	0.53	0.47	0.93	0.62

the senses of a word. CIA* labels a phrase as metaphorical if any of a target word's synsets exist in the list of synsets that belong to the collocated nouns. The pseudocode for our CIA* algorithm is presented in Algorithm 7.

```
Algorithm 7: CIA* Algorithm
Input: Sentence, Verb, Object Noun
Output: Metaphorical classification as true or false
1: Query Bing API for top 50 Web sites related to Verb
2: Extract sentences from the Web sites' content that contain Verb
3: Parse each sentence to find collocated nouns
4: Use WordNet to find the synsets and hyponyms for all of the collocated nouns and the synsets of
   Object Noun
5: if the main synset does not appear in the list of synsets related to the collocated nouns then
6:     return true
7: else
8:     return false
9: end if
```

4.2 Combined approach (CIAC)

This variation of the CIA* algorithm considers concreteness ratings of the target noun as a pre-processing step. The algorithm first looks at the concreteness rating of the target noun and the metaphor candidate is classified as metaphorical if its concreteness value is below the threshold otherwise, the metaphor candidate is further processed by the CIA* algorithm.

A threshold value was set to 4.0. To identify an appropriate threshold value, the abstractness ratings of target nouns in the test dataset were collected from the pre-annotated TroFi Example Base and considered the median concreteness ratings of the nonliteral and literal annotations.

Brysbaert, Warriner, and Kuperman's list of "Concreteness ratings for 40 thousand generally known English word lemmas" was used [40] as the source of concreteness ratings. The list was developed for researchers who require the concreteness ratings of words [40]. Data was collected via crowdsourcing from more than four thousand participants. Participants rated words on a scale from 1 (abstract) to 5 (concrete) [40]. The result is a dataset of 37,058 English words and 2,896 two-word expressions with concreteness values between 1 and 5 [40].

4.3 Concreteness model (CON)

This approach is essentially the first half of CIAC; it bases its decision about whether a phrase is metaphorical solely on the concreteness score of the target noun. This approach was icluded in the present evaluation to compare this simple implementation of the Concrete-Abstract theory with other existing variations and to measure the number of false negatives in comparison with CIAC.

Again, the list of concreteness ratings [40] was used as the source of concreteness scores. If the concreteness score of the target noun is less than some threshold t, the algorithm classifies the candidate as metaphorical, else

Table 3 Binary classifications and results: simple data set

	True	False	True	False	No	Results			
	Pos.	Pos.	Neg.	Neg.	Decision	Acc.	Pre.	Rec.	F-score
CIA+	19	13	2	3	2	0.54	0.59	0.86	0.69
CIA*	17	10	7	5	0	0.61	0.62	0.77	0.61
CIAC	16	6	11	6	0	0.69	0.72	0.66	0.68
CON	20	14	3	2	0	0.58	0.58	0.90	0.75

Table 4 Comparison with other metaphor identification systems

	Dataset	Accuracy	Precision	Recall	F-score
Turney et al. [10]	TroFi [42]	0.688	NR	NR	0.681
Birke and Sarkar [42, 44]	TroFi	NR	NR	NR	0.649
Dunn [21]	VU Amsterdam [45]	NR	NR	NR	0.792
CCO* (Neuman et al. [13])	Reuters RCV1 [46]	NR	0.761	0.82	NR
CIA+	TroFi	0.54	0.59	0.86	0.69
CIA*	TroFi	0.61	0.62	0.77	0.61
CIAC	TroFi	0.69	0.72	0.66	0.68
CON	TroFi	0.58	0.58	0.90	0.75

the metaphor candidate is classified as literal. If the target word does not exist in the list of concreteness ratings [40], the algorithm cannot decide whether the candidate is metaphorical. The pseudocode for the CON model is provided in Algorithm 8.

```
Algorithm 8: CON Algorithm
Input: Noun
Output: Metaphorical classification as true or false
1: Lookup concreteness score for noun in Brysbaert's list of concreteness ratings
2: if score < 4.0 then
3:     return true
4: else
5:     return false
6: end if
```

5 Experimental study for performance comparison with the state-of-the-art

This section reports the results of an experiment through which the performance of the CIA+, CIA*, and CIAC are compared. The experiment is conducted based on the TroFi dataset. The experiment evaluates the present algorithm with test data and reports the binary classifications as well as the accuracy, precision, recall, and F-score of each variation.

5.1 Experimental procedure and setup

The CIA*, CIA+, and CIAC models all require WordNet as an external knowledge source. The WordNet corpus included in the NLTK library [41] was used. Both CON and CIAC require concreteness scores for nouns. The values

Table 5 Binary classifications and results: phishing data set

	True	False	True	False	No	Results			
	Pos.	Pos.	Neg.	Neg.	Decision	Acc.	Pre.	Rec.	F-score
CIA+	0	1	44	35	0	0.44	0.00	0.00	0.00
CIA*	0	1	38	41	0	0.51	0.00	0.00	0.00
CIAC	0	0	53	26	0	0.33	0.00	0.00	0.00
CON	0	1	35	44	0	0.55	0.00	0.00	0.00

from Brysbaert, Warriner, and Kuperman's list of concreteness ratings [40] were utilized. The dataset consists of 37,058 English words that have concreteness values from 1 (abstract) to 5 (concrete).

5.2 TroFi data set

Metaphor identification research struggles to make progress because of a lack of shared resources and reporting methods [7]. The present research evaluated the present algorithm with Birke and Sarkar's TroFi dataset as input because it is well known in the field and thus provides a standard for comparison.

The publicly available, highly accessible, TroFi Example Base provided by Birke and Sarkar [42] was used as input for the evaluation of our algorithms. To maintain a dataset with a balanced number of nonliteral and literal sentences, a group of literal sentences and a group of nonliteral sentences was utilized. Then, a random sample of 50 sentences from each group was selected. Of the 100 sentences selected for evaluation, only 71 had at least one direct object relation, leaving 34 nonliteral sentences and 37 literal sentences for the evaluation of the present algorithm.

5.3 Assessment metrics

For a comprehensive understanding of the algorithm's performance, the following are reported: the binary classifications, as well as accuracy, precision, recall, and F-score. The binary classifications are defined as follows:

- *True Positive* is the correct classification of metaphor as metaphor.
- *True Negative* is the correct classification of non-metaphor as non-metaphor.
- *False Positive* is the incorrect classification of a non-metaphor as a metaphor.
- *False Negative* is the incorrect classification of a metaphor as a non-metaphor.

Furthermore, 1) *Accuracy* measures the correct classification rate, 2) *Recall* is the ability to find all the metaphors in the dataset, 3) *Precision* expresses the proportion of data that is classified as metaphorical versus the data that is metaphorical, and, 4) *F-score* describes accuracy as a weighted mean of precision and recall. The formulas for calculating these metrics are given below:

$$Accuracy = \frac{TruePositive + TrueNegative}{(FalsePositive + FalseNegative}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegative}$$

$$Precision = \frac{TruePostitives}{TruePositives + FalsePositives}$$

$$F - score = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

5.4 Implementation details

To evaluate the present algorithms, a software application that implements all four algorithms was built. The application is written in C# and python. Integrating two programming languages was necessary to utilize existing NLP tools that are written in python.

The WordNet::Similarity package [43] was used to find the synsets and hyponyms of all of the collocated nouns. The WordNet::Similarity package [43] is part of a popular NLP toolkit, NLTK, that contains functions to explore WordNet as a corpus. An existing python implementation of the Adapted Lesk Algorithm [20] provided by pywsd was used in the present implementation.

In addition to the binary classifications, the software program also returned the WordNet senses of the noun from the WordNet::Similarity package [43] and the single synset returned from the pywsd implementation of the Adapted Lesk Algorithm [41]. Each input sentence was run individually through the software application that outputs the results of each approach and then copy and pasted the results into a spreadsheet, which contained the sentence and its original annotation from the TroFi data [42]. The spreadsheet contained the raw data necessary for the empirical evaluation.

Table 6 Phishing: verbs that were identified as metaphorical

Induced, targets, reaching, trusting, block, uses, used, access, contains, used, avoid, embedding, altered, roll, redirect, sent, used, sent, exchanged

5.5 Observed results

5.5.1 Full data set

Table 2 shows the results for the four different approaches. In CIA* and CIAC, the positive predictions outnumber the negative predictions. While the number of false negatives in the word sense disambiguation variation is lower than those in the class inclusion and concreteness models, it is higher than the number of false positives predicted by CIAC. Despite a decrease in overall accuracy, adding word sense disambiguation results in higher recall, suggesting that the metaphors it does identify are more relevant.

In some instances, the sentence contained more than one direct object relation. The software application handles such cases and returns results for every direct object relation in the sentence. In the spreadsheet, each direct object relation was treated as a separate sentence; a new data point was created and considered the human annotation of the original sentence to be the annotation for each new data point. The application returned 107 results for the 71 distinct input sentences.

5.5.2 Simple data set

To examine the impact of multiple direct object relations in a sentence on the output of the algorithms, only the sentences that contain a single direct object relation were evaluated. This smaller input set is referred to here as the "Simple" dataset. The Simple dataset consisted of 39 distinct sentences; 22 nonliteral and 17 literal. Table 3 shows the results of the experiment using the Simple dataset.

In both the Full and Simple datasets, the concreteness model has the lowest recall. The accuracy of all variations increases with the Simple dataset; however, each variation performs the same in relation to the other regardless of the input.

5.6 Data analysis

The results presented in Table 3 suggest that CIA+ performed worse than CIA* alone. This suggests that adding word sense disambiguation did not reduce the rate of false negatives as expected. However, adding word sense disambiguation did increase recall. The accuracy of metaphor identification was higher on the Simple dataset. This implies that not every part of a sentence must be nonliteral for a sentence to be considered nonliteral.

5.7 Comparison with the state-of-the-art

Table 4 compares the performance of the present methodology with other algorithms from the literature that use the same or similar datasets as input. More specifically, the table presents data from research published by 1) Turney et al. [10], Birke and Sarkar [44], Dunn [21], and Neuman et al. [13].

According to Table 4, CON outperforms the Concrete-Abstract algorithm (Turney et al. [10]) in terms of F-score, but falls short of meeting the accuracy level achieved by this algorithm. More specifically, the CON approach achieves 0.58 and 0.75 for accuracy and F-score; whereas, Turney et al. report the accuracy of 0.688 and F-score 0.681. This observation provides more information about using context to identify metaphor. Turney et al. measures the abstractness of the sentence by calculating the abstractness of every word in the sentence [10]. The CON model bases its decision about whether a phrase is metaphorical based on the concreteness value of the target noun alone. The accuracy of their system suggests that computing values for every word in the entire sentence may not be necessary.

Similarly, the CON methodology outperforms Birke and Sarkar's [42, 44] work, but is slightly short of meeting the F-score of 0.792 set by Dunn [21]. Unfortunately, these research studies do not report the entire assessment and performance metrics and their values. Therefore, only F-scores are used for a meaningful comparison between the CON methodology and the existing work.

6 Applications of metaphor detection in cyber security documents

In this section, three case studies are presented. Each case study focuses on a different cybersecurity threat: 1) phishing, 2) denial-of-service and 3) information leakage. The case studies presented here demonstrate the algorithms' performance on texts about cybersecurity threats.

6.1 Research objectives

The high-level research question is: How does the present algorithm perform on text about cybersecurity threats? From the case studies, the following will be accomplished:

Table 7 Binary classifications and results: denial-of-service data set

	True	False	True	False	No	Results			
	Pos.	Pos.	Neg.	Neg.	Decision	Acc.	Pre.	Rec.	F-score
CIA+	3	60	16	2	0	0.23	0.05	0.60	0.09
CIA*	2	46	32	3	0	0.41	0.04	0.40	0.08
CIAC	3	69	15	2	0	0.20	0.04	0.60	0.08
CON	3	53	25	1	0	0.34	0.05	0.75	0.10

1. describe the input text extracted from a broad search of the sources on the Internet;
2. evaluate the effectiveness of the present metaphor identification system on texts specific to the cybersecurity threats;
3. observe the intermediate data to aid in the understanding of the system output.

To do so, three different types of cyber attacks are examined: 1) phishing, 2) denial-of-service, and 3) information leakage. These types of attacks were chosen based on the assumption that phishing is very prevalent problem but hard to use metaphor in this context and it is ambiguous; whereas, a denial-of-service attack is very intuitive and a simple metaphor can describe it. On the other hand, information leakage might be a concept in the middle borrowing metaphor from law, business, and technology.

6.2 Methodology

To collect the input data for evaluation, a Web scraper built specifically for this project was used. The Web scraper takes the name of the cybersecurity threat as input and outputs a list of sentences from the Web that contain the name of the cybersecurity threat. The output of the Web scraper is the input data for the present algorithm.

To empirically evaluate the output of the present system, the input data was annotated so that we could compare the output of the algorithms with human judgment. Annotations were done by two graduate students. The annotators labeled the input sentences as literal or nonliteral. If the sentence was labeled nonliteral, the annotator was asked to identify which words in the sentence they considered nonliteral and then to select the type of nonliteral language from a list of nonliteral language types provided to them in the task instructions.

To establish credibility, inter rater agreement, a measure of reproducibility of human judgment [47], had to be sufficiently high. Guilford's G [48] was employed, instead of Cohen's k [49], as the measure for inter-rater agreement because Cohen's k is sensitive to distributional skew [50] and the present dataset contained significantly more literal than nonliteral sentences.

G > 0.7 was set as a threshold to accept the annotations. After the first round of annotations, the inter-rater reliabilities for phishing, denial-of-service, and information leakage were 0.76, 0.80, and 0.60 respectively. The inter-rater reliability was high enough for phishing and denial-of-service but not for information leakage. After the first round, the annotators revisited their decisions until they reached agreement on each of the sentences. Because the present focus is on metaphor identification, annotations marked as nonliteral but not metaphorical, were considered to be false.

In the following sections, each case study and the results are explained.

6.3 Case study I: phishing (very ambiguous nonliteral)

The first case study concerns the cybersecurity threat "phishing".

6.3.1 Input data

The word "phishing" was used as inputted into the Web scraper. The Web scraper outputs 50 sentences that contain the word "phishing". By observing the annotators' ratings and comments, it was observed that out of the 50 sentences, only 4 were annotated as nonliteral, and of those, only 2 were annotated as metaphorical.

6.3.2 Empirical analysis

From the 50 input sentences, the algorithms identified 80 direct object relations. Results of the empirical analysis are shown in Table 5. All four variations of the algorithm perform poorly on this unbalanced dataset. The precision, recall, and F-score are all 0.00 because there is only 1 Type

Table 8 Direct object relations

Verb	Direct object
Overwhelm	It
Prevent	Users
Accessing	Website

Table 9 Binary classifications and results: information leakage data set

	True	False	True	False	No	Results			
	Pos.	Pos.	Neg.	Neg.	Decision	Acc.	Pre.	Rec.	F-score
CIA+	1	0	47	14	0	0.24	0.02	1.00	0.04
CIA*	0	1	43	18	0	0.29	0.00	0.00	0.00
CIAC	1	0	52	10	0	0.17	0.02	1.00	0.04
CON	1	0	40	21	0	0.35	0.02	1.00	0.05

II metaphor in the dataset and the algorithm classifies it incorrectly.

6.3.3 Observations

In the following, some of the intermediate data are examined to identify trends in the output and explain some of the misclassifications. Table 6 shows a list of verbs from the sentences identified as metaphorical by all four of the algorithms. These are the verbs in the direct object relations that are incorrectly classified as metaphorical by the algorithms.

One of the major problems with selectional preference violation is that it tends to classify common verbs as metaphorical. As such, it is not surprising that the verb "use", a common verb, shows up several times in the list of verbs that are incorrectly classified as metaphorical. However, the verb "induced" is not as common as "use" and it still shows up in our list of verbs. To understand why, the sentence that contains the verb "induced" was identified, and it was ran through the software application in debug mode to catch some of the intermediate data. Observations from the application are outlined below.

Analyzing the Intermediate Data. To observe intermediate data, the following sentence was used:

> *"A phishing email to Google and Facebook users successfully induced employees into wiring money to the extent of US 100 million to overseas bank accounts under the control of a hacker"*

as input into our application.

(I) The first step of the present algorithm is to identify direct object relations by parsing the input sentence using the Stanford Dependency Parser [38]. The results of the parse suggest that the direct object relation the present algorithm is evaluating contains "induced" as the verb and "employees" as the direct object noun.

(II) Next, the verb "induced" gets fed into the real time corpus builder that creates a narrow corpus for identifying semantic relations by extracting text from the top Web sites that the Bing API returns from the query "example usages of induced". The corpus builder returns a list of sentences which contain the word "induced" from the list of Web sites returned from the Bing API. The list contains several dictionary Web sites (https://www.yourdictionary.com, https://www.merriam-webster.com, https://www.dictionary.com, https://www.thesaurus.com) that are assumed to have examples of the different senses of the word "induced". There are also several Web sites assumed to include technical language (https://www.stemcells.nih.gov, https://www.ncbi.nlm.nih.gov, https://www.researchgate.net).

(III) Next, nouns collocated with the verb "induced" are extracted from the Web sites. This is done by scraping the contents of each Web site and extracting every sentence that contains the word "induced". Each sentence is then parsed by the Stanford Dependency Parser [38]. Nouns are considered collocated if they appear in a direct object relation with the verb "induced". The collocated nouns for the term "induced" the system returns are: self, anxiety, interaction, diipoles, dipole-dipole, him, Statistics, us, labor, compromise, Scrabble, wind, magnetsim, magnetism, it, therapy, demand, current, emission, cells, trip, many, girl, man, me, article, Ossipon, Mr., voltage, Verb, Examples, C., indubitables, disorders, disorder, lysate, sample.

(IV) The word "Scrabble" shows up in the list of collocated nouns, this happens because the game of "Scrabble" is mentioned frequently in the HTML markup from https://www.merriam-webster.com/dictionary/induce. This suggests that the process of cleaning the HTML markup could be improved. V) Summary: The following sentence was chosen:

> *"A phishing email to Google and Facebook users successfully induced employees into wiring money to the extent of US 100 million to overseas bank accounts under the control of a hacker'*

through the software application in debug mode to extract intermediate data because it was unexpected that

Table 10 Results: information leakage data set

	Accuracy	Precision	Recall	F-score
CIA+	0.24	0.02	1.00	0.04
CIA	0.29	0.00	0.00	0.00
CIAC	0.17	0.02	1.00	0.04
CON	0.35	0.02	1.00	0.05

a less frequently used verb, such as "induced" was marked as metaphorical by all four algorithms.

(VI) From the false positive output, it was observed that frequent verbs, such as "use", are misclassified. To reduce the number of false positives, it might be worth identifying the frequency of verb usage in the English language and setting a threshold to automatically classify direct object relations that contain verbs that are used frequently as false.

6.4 Case study II: denial-of-service (clear non-literanl)

The second case study concerns the cybersecurity threat, "denial-of service".

6.4.1 Input data

The words "denial-of-service", "denial-of-service" and "DoS" were used as input into the web scraper, which resulted in 52 sentences for evaluation. Eight of the 52 sentences were annotated as metaphorical by our annotators. All eight sentences labeled as metaphorical by our annotators contain the word "flood". In seven sentences, the word "flood" was used as a verb, and in the other, "flood" was used as a noun.

6.4.2 Empirical analysis

From the 52 sentences, the algorithms identified 91 direct object relations. The results from the empirical analysis are shown in Table 7.

6.4.3 Observations

Next, the results are examined as a set to identify trends in the output. The annotators identified seven sentences with the verb "flood", the algorithms only identified five sentences that contain the verb "flood". To understand why, one of the two sentences not identified by the algorithms was identified and ran through the software application in debug mode to catch some of the intermediate data. The following sentence was entered into the software application for observation:

> *"A DoS, or denial-of-service attack, floods a system, often a web server, with data in order to overwhelm it and prevent users from accessing a website".*

During the pre-processing step, the dependency parser returned the three direct object relations seen in Table 8. The direct object relation that the annotators marked as nonliteral was the verb-noun pair (flood, system). In this case, the algorithm fails because the parser fails to return the direct object relation of interest.

Additionally, the parser returns the direct object relation (overwhelm, it). The word "it" is actually a pronoun, not a noun. The parser is not incorrect because the word "it" refers to the noun "web server". When processing the input sentence, the word "it" is input into WordNet to find the sense of the word. WordNet interprets the pronoun "it" as the acronym "IT" and returns "information_technology.n.01". This is a prime example of the issue of pronouns in selectional preference violation. Because collocates are identified by their part of speech, a noun, any direct object relation that includes the pronoun "it", is going to be classified as metaphorical.

6.4.4 Summary

In this case study, the following sentence was chosen:

> *"A DoS, or denial-of-service attack, floods a system, often a web server, with data in order to overwhelm it and prevent users from accessing a website."*

through the software application in debug mode to watch the intermediate data. This sentence was chosen because it was one of two sentences that the algorithms should identify as metaphorical because it contains the verb "flood". It was discovered that the direct object relation that contains the verb was not identified by the parser. As a result, the relation was not even considered by the algorithms. This demonstrated the impact of the parser on the overall accuracy of the algorithms. Additionally, the issue of pronouns in selectional preference violation was observed.

6.5 Case study III: information leakage (confusing nonliteral)

The final case study concerns the cybersecurity threat, "information leakage".

6.5.1 Input data

Originally, the phrase "information disclosure" was used as input into the web scraper to find sentences for this cybersecurity threat. However, most of the sentences returned from the web scraper discussed the legal ramifications of information disclosure, not a cybersecurity threat, i.e., the content was in the wrong domain. Accordingly, the input phrase was changed to "information leakage" and the sentences that were returned belonged largely to the cybersecurity domain. Even then, many of the top sentences were related to vendors or vendor services rather than "information leakage" as a threat. Most sentences relating to vendors were removed from the input dataset because the present aim concerned text about cybersecurity threats, not products.

Of the 51 sentences identified as input data for "information leakage", the annotators only labeled 1 sentence as metaphorical. Another five sentences were labeled nonliteral but typed as personification Table 9.

6.5.2 Data analysis

The results from the empirical analysis are captured in Table 10.

6.5.3 Observations

Out of the 51 sentences, 5 did not contain any direct object relations. Out of the remaining 46 sentences, the algorithms identified 63 direct object relations.

Unlike the other case studies, several sentences in the "information leakage" dataset caused an exception (the filename or extension is too long) in the software application. In one of those cases, the sentence was short. Intuitively, it seems that an overflow would be due to a longer sentence. The following short sentence, which resulted in an exception, was run through the system in debug mode to understand what happened: *" Designers of secure systems often forget to take information leakage into account."*

First, the parser returned the direct object relation (take, leakage). Next, the collocate extraction returned 1,228 nouns. Then the system found synsets and hyponyms of each noun. It is at this step that the exception, "filename too long" occurs because one of the sentences extracted from the Internet contained more than 40,000 characters.

WordNet::Similarity, an open source NLP software tool written in python [43], is used to find the synsets of a noun. Because the present program is written in C#, it is limited to passing information to the python program via a hidden command prompt using a json string. The max character limit of the command prompt is 8,191 characters, so the 40,000-character sentence causes an exception. During development, the 8,191-character limitation seemed inconsequential because single sentences are not that long.

As a temporary solution, the sentence length was truncated at 6,000 characters to prevent an overflow and it was ran again for observation. The 40,000-character string of text comes from the dictionary entries of the word "take" from https://www.yourdictionary.com/take. This happens because there are many definitions of the word "take" and not all are written in complete sentences. The web page contains more than 60 definitions of the word "take". This experience revealed that one cannot assume that text on web pages is written in complete sentences.

6.5.4 Summary

The behavior of the algorithms on "information leakage" data was different from the "phishing" and "denial-of-service" case studies. The output of the software application reveals that approximately 10% of the input sentences did not contain any direct object relation. Around 20% of the direct object relations contained the threat name itself; the target noun being "leakage". This suggests that the language used to speak about information leakage is different from the language about phishing or denial-of-service.

6.5.5 Concluding remarks

In these case studies the input text was described, performance of the algorithm on text about cybersecurity threats was evaluated, and intermediate data of the software application was observed. The numerical results suggest that, for some reasons, the algorithm does not perform well on the present unbalanced datasets that contain mostly literal language. These case studies demonstrate the impact of unbalanced datasets on the reported accuracy of the metaphor identification algorithm.

The observations of the annotators' ratings and comments suggest that there are few metaphors in the dataset scraped from the Web. To find metaphors, it will be necessary to search beyond the top results returned from the Bing API. Perhaps updating the web scraper's query from a single word to return Web sites from specific topics (academic, news, images) or pulling data from other sources such as social media, would increase the number of metaphors in the input data.

To explain the output and help identify areas of improvement for future work, intermediate data were observed as the software application ran in debug mode. By doing this, it was demonstrated that the algorithm suffers in the same ways that other algorithms based on selectional preference violation struggle; with pronoun resolution, common language bias and polysemy.

These case studies show that empirical analysis alone is not indicative of an algorithm's ability to identify metaphors related to text about cybersecurity threats. The major takeaway is that the desired metaphorical language is not found in the text from top search results. The next step then, is searching for text that may contain metaphors.

7 Conclusions and future work

Metaphor is a method used to communicate about abstract concepts. It is a topic of study in many different fields including cognitive psychology, linguistics, natural language processing and cybersecurity. The wide variety of metaphor identification approaches, many dependent upon external knowledge sources, report varying results suggesting that the metaphor identification task is not simple. Moreover, it is difficult to compare existing approaches because of the lack of a standard reporting mechanism and different evaluation datasets.

Metaphor has been gaining attention in the field of cybersecurity [5, 6]. Studies show that including metaphor in user interfaces increases a user's understanding of a security concept and influence their online behavior [5]. Automatically identifying metaphor in cybersecurity lays the groundwork for the development of software that effectively communicates the risk of cybersecurity threats to end-users.

7.1 Contributions

To contribute to natural language processing, specifically metaphor identification, four metaphor identification algorithms were designed and implemented, and their performance was compared to one another and to that of existing algorithms that use the same or very similar test data. The simplified versions perform comparably. More specifically, a methodology, CIA+, and several variations, were compared to evaluate the impact of word sense disambiguation on metaphor identification. The CIA+ algorithm was compared with existing approaches that are designed to identify Type II metaphors and that use the same, or similar, data to observe any differences.

To address the problem of polysemy in metaphor identification, a methodology that integrates word sense disambiguation was created. Results revealed that considering the context of a sentence, rather than just the word or phrase alone does not increase the accuracy of a system. Additionally, the same accuracy was achieved by simply evaluating the concreteness score of the target noun similar to the well-known Concrete-Abstract algorithm by Turney et al. [10]. The difference between the two approaches is that the Concrete-Abstract approach considers the abstractness of all the words in a sentence and the present approach only considers the target noun.

A real time corpus builder was developed and integrated to eliminate the need for large knowledge sources sometimes necessary in existing metaphor identification systems. There are several benefits of creating a corpus from the Web. First, the knowledge source for identifying normal language usage will always be current and contain cybersecurity words that may not exist in large corpora. Second, using the Web makes the metaphor identification system usable on a personal computer. Rather than having to store and search a large dataset, the Web is used to find collocates. Finally, the real time corpus from the Web provides unique opportunities for future work, which are discussed in the next section.

To get a sense of the type and amount of metaphors available in the text related to specific cybersecurity threats, case studies were performed on the following threats: denial-of-service, phishing, and information leakage. Input data specific to individual cybersecurity threats was gathered by scraping text from the top Web sites returned from the Bing API. The case studies reveal that very few metaphors appear in text in the top Web search results, and that the present algorithm reports many false positives.

7.2 Future work

The real time corpus eliminates the need for large preparsed corpora to find collocates. This helps overcome limitations imposed by the size of corpora [13]. Currently, the present real time corpus only returns the top 50 Web pages, but it could be modified to continue searching until the number of desired collocates is found.

The Stanford Dependency Parser [38] was used to identify the grammatical structure of a sentence. Although this parser is well known and widely used, it produces errors, which impacts the accuracy of our algorithm. Evaluating the impact of different parsers on the accuracy of the algorithm is an area for future study.

7.2.1 Cybersecurity

The present research revealed from the annotations done on the input data for the case studies around certain cybersecurity threats that there are only a few metaphorical sentences that contain the cybersecurity threat. It was also observed that the present algorithm classifies many sentences as metaphorical when they are not. Future areas of research from these observations include searching for text that may contain metaphor and enriching knowledge sources to include text about cybersecurity.

Searching different data sources for more abundant use of metaphor: This can be done by modifying the query in the Bing API to return different results, perhaps targeted at news articles, academic journals or even image descriptions. Different input data could also be scraped from social media sites such as Facebook or Twitter.

To reduce the number of false positives, one could build a new, or enrich an existing, knowledge source. In the present research, WordNet was used as a knowledge base to create ad-hoc semantic classes. WordNet did not always contain the object noun in the direct object relation. This is a cause for misclassification because if a noun does not exist in WordNet, the metaphor candidate is automatically classified as metaphorical. Expanding WordNet to include cybersecurity text would help reduce the number of false positives that are caused by a lack of hand-coded knowledge.

7.2.2 Application

The software application developed for the present research was designed to test the feasibility of a small-scale metaphor identification algorithm. It was written in C#. However, numerous NLP tools are written in python and some of the required workarounds introduced limitations. Because of that, it might be more efficient to rewrite the application in python for compatibility. The software application was written so that different grammar relation rules can be applied easily. Finally, it might be useful for the sake of data exploration to expose some of the intermediate data to observe the nouns that are identified as collocates. Searching different domains (medical, art, cancer, etc.) or different types of media (journal articles, dictionaries, social media, etc.) could potentially return a different set of collocated nouns and might give insight to the where data is located.

7.2.3 Extension of larger project

This work was part of a greater research effort to communicate the risk of cybersecurity threats to visually impaired end users. One of the challenges in that research is identifying auditory cues for effective communication. Right now, choosing sounds to convey meaning involves human judgment. This work is a step towards automating the process. Toward that end, the present research developed a tool to identify metaphors, which could be useful for user interface design. Assuming this tool is sufficient for identifying metaphor candidates, these are the next steps:

1. mapping the verbs from the identified Type II metaphors to auditory cues;
2. and, evaluating the effectiveness of communicating cybersecurity threats through metaphor.

From this work emerges a tool usable for future research and insight into both, metaphor identification systems and cybersecurity data. The present findings inspire many new directions for future research in both metaphor identification and cybersecurity. The evaluations and observations from this research demonstrate that while a lot of research has been done on metaphor identification, there is still a long way to go.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Lakoff G, Johnson M (2008) Metaphors we live by.University of Chicago press
2. Lakoff G (1994) Master metaphor list. University of California
3. Shutova E, Sun L, Korhonen A (2010) Metaphor identification using verb and noun clustering. In: Proceedings of the 23rd international conference on computational linguistics, ser. COLING '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp 1002–1010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1873781.1873894
4. Camp LJ (2009) Mental models of privacy and security. IEEE Technol Soc Mag 28(3):37–46
5. Raja F, Hawkey K, Hsu S, Wang K-LC, Beznosov K (2011) A brick wall, a locked door, and a bandit: a physical security metaphor for firewall warnings. In: Proceedings of the 7th symposium on usable privacy and security. ACM, 2011, p 1
6. Moore JH, Parrott LK, Karas TH (2008) Metaphors for cyber security. SANDIA. Tech, Rep
7. Shutova E (2015) Design and evaluation of metaphor processing systems. Comput Linguist 41(4):579–623
8. Lakoff G (1993) The contemporary theory of metaphor. 0
9. Krishnakumaran S, Zhu X (2007) Hunting elusive metaphors using lexical resources," in Proceedings of the Workshop on Computational approaches to Figurative Language. Assoc Comput Linguist, 13–20
10. Turney PD, Neuman Y, Assaf D, Cohen Y (2011) Literal and metaphorical sense identification through concrete and abstract context. In: Proceedings of the conference on empirical methods in natural language processing. Association for computational linguistics, 2011, pp 680–690
11. David O, Matlock T (2018) Cross-linguistic automated detection of metaphors for poverty and cancer. Lang Cogn 10(3):467–493
12. Gandy L, Allan N, Atallah M, Frieder O, Howard N, Kanareykin S, Koppel M, Last M, Neuman Y, Argamon S (2013) Automatic identification of conceptual metaphors with limited knowledge. AAAI 2:1–1
13. Neuman Y, Assaf D, Cohen Y, Last M, Argamon S, Howard N, Frieder O (2013) Metaphor identification in large texts corpora. PLoS One 8(4):e62343
14. Haagsma H, Bjerva J (2016) Detecting novel metaphor using selectional preference information. In: Proceedings of the fourth workshop on metaphor in NLP, 2016, pp 10–17

15. Mason ZJ (2004) Cormet: a computational, corpus-based conventional metaphor extraction system. Comput Linguist 30(1):23–44
16. Resnik P (1997) Selectional preference and sense disambiguation. Tagging Text with Lexical Semantics: Why, What, and How?, 1997
17. Fass D, Wilks Y (1983) Preference semantics, ill-formedness, and metaphor. Comput Linguist 9(3–4):178–187
18. Fellbaum C (1998) Wordnet: an electronic lexical database and some of its applications
19. Lesk M (1986) Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: Proceedings of the 5th annual international conference on Systems documentation. ACM, 1986, pp 24–26
20. Banerjee S, Pedersen T (2002) An adapted lesk algorithm for word sense disambiguation using wordnet. In: Proceedings of the international conference on intelligent text processing and computational linguistics. Springer, 2002, pp 136–145
21. Dunn J (2013) 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
22. Gao G, Choi E, Choi Y, Zettlemoyer L (2018) Neural metaphor detection in context. In arXiv preprint arXiv:1808.09653, p 2018
23. Gong H, Gupta K, Jain A, SumaBhat (2020) Illinimet: illinois system for metaphor detection with contextual and linguistic information. In: Proceedings of the second workshop on figurative language, 2020, pp 146–153
24. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) Roberta: a robustly optimized bert pretraining approach. arXiv:1907.11692
25. Mao R, Lin C, Guerin F (2019) End-to-end sequential metaphor identification inspired bylinguistic theories. In: Proceedings of the 57th annual meeting of the association for computational linguistics, 2019, pp 3888–3898
26. Steen GJ, Dorst AG, Hermann JB, Kaal A, abd Trijntje Pasma TKK (2010) A method for linguistic metaphor identification: From mip to mipvu. John Benjamins Publishing, vol. 14
27. Wilks Y, Fass D (1992) The preference semantics family. Comput Math Appl 23(2–5):205–221
28. Stowe K, Moeller S, Michaelis L, Palmer M (2019) Linguistic analysis improves neural metaphor detection. In: Proceedings of the 23rd conference on computational natural language learning (CoNLL), 2019, pp 362–371
29. Stowe K, Palmer M (2018) Leveraging syntactic constructions for metaphor identification. In: Proceedings of the workshop on figurative language processing, 2018, pp 17–26
30. Su C, Fukumoto F, Huang X, Li J, Wang R, Chen Z (2020) Deepmet: A reading comprehension paradigm for token-level metaphor detection. In: Proceedings of the second workshop on figurative language processing, 2020, pp 30 – 39
31. Klebanov BB, Leong CWB, Flor M (2018) A corpus of non-native written english annotated for metaphor. In: Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: human language technologies, vol. 2, pp. 86 – 91, 2018
32. Chakrabarty T, Zhang X, Muresan S, Peng N (2021) Mermaid: metaphor generation with symbolism and discriminative decoding. In: Proceedings of the annual conference of the north american chapter of the association for computational linguistics (NAACL), 2021
33. Gero KI, Chilton LB (2019) Metaphoria: an algorithmic companion for metaphor creation. In: Proceedings of the 2019 CHI conference on human factors in computing systems, 2019, pp 1–2
34. Stowe K, Ribeiro L, Gurevych I (2020) Metaphoric paraphrase generation. In arXivpreprint arXiv:2002.12854
35. Davies M (2010) The corpus of contemporary american english as the first reliable monitor corpus of english. Lit Linguist Comput 25(4):447–464
36. Research P Wordnet based categorization dictionary. https://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/wordnet-based-categorization-dictionary/
37. Speer R, Havasi C (2013) Conceptnet 5: A large semantic network for relational knowledge. In The People's Web Meets NLP. Springer, 2013, pp 161–176
38. Schuster S, Manning CD (2016) Enhanced english universal dependencies: an improved representation for natural language understanding tasks. In LREC. Portorož, Slovenia, 2016, pp 23–28
39. Glucksberg S, Keysar B (1990) Understanding metaphorical comparisons: beyond similarity. Psychol Rev 97(1):3
40. Brysbaert M, Warriner AB, Kuperman V (2014) Concreteness ratings for 40 thousand generally known english word lemmas. Behav Res Methods 46(3):904–911
41. Bird S, Klein E, Loper E (2009) Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc.
42. Birke J, Sarkar A (2006) A clustering approach for nearly unsupervised recognition of nonliteral language. In: Proceedings of the 11th conference of the european chapter of the association for computational linguistics, 2006
43. Pedersen T, Patwardhan S, Michelizzi J (2004) Wordnet: similarity: measuring the relatedness of concepts, in Demonstration papers at HLT-NAACL. Assoc Comput Linguist 2004:38–41
44. Birke J, Sarkar A (2007) Active learning for the identification of nonliteral language. In: Proceedings of the workshop on computational approaches to figurative language. association for computational linguistics, 2007, pp 21–28
45. Steen G, Herrmann ADJ, Kaal A, Krennmayr T (2010) Metaphor in usage. Cognitive Linguist 21(4):765–796
46. Lewis D, Yang Y, Rose T, Li F (2004) Rcv1: a new benchmark collectionfor text categorization research. J Mach Learn Res 5:361–397
47. Xu Y, Malt BC, Srinivasan M (2017) Evolution of word meanings through metaphorical mapping: systematicity over the past millennium. Cogn Psychol 96:41–53
48. Hovy D, Shrivastava S, Jauhar SK, Sachan M, Goyal K, Li H, Sanders W, Hovy E (2013) Identifying metaphorical word use with tree kernels. In: Proceedings of the first workshop on metaphor in NLP, 2013, pp 52–57
49. Cohen J (1960) A coefficient of agreement for nominal scales. Educ Psychol Measur 20(1):37–46
50. Xu S, Lorber MF (2014) Interrater agreement statistics with skewed data: Evaluation of alternatives to cohen's kappa. J Consult Clin Psychol 82(6):1219