machine learning & knowledge extraction

MDPI

Article

Leaving No Stone Unturned: Flexible Retrieval of Idiomatic Expressions from a Large Text Corpus

Callum Hughes [1], Maxim Filimonov [1], Alison Wray [2] and Irena Spasić [1,*]

[1] School of Computer Science and Informatics, Cardiff University, Cardiff CF24 3AA, UK; hughescj4@cardiff.ac.uk (C.H.); FilimonovM@cardiff.ac.uk (M.F.)
[2] School of English, Communication and Philosophy, Cardiff University, Cardiff CF10 3EU, UK; WrayA@cardiff.ac.uk
* Correspondence: spasici@cardiff.ac.uk

Abstract: Idioms are multi-word expressions whose meaning cannot always be deduced from the literal meaning of constituent words. A key feature of idioms that is central to this paper is their peculiar mixture of fixedness and variability, which poses challenges for their retrieval from large corpora using traditional search approaches. These challenges hinder insights into idiom usage, affecting users who are conducting linguistic research as well as those involved in language education. To facilitate access to idiom examples taken from real-world contexts, we introduce an information retrieval system designed specifically for idioms. Given a search query that represents an idiom, typically in its canonical form, the system expands it automatically to account for the most common types of idiom variation including inflection, open slots, adjectival or adverbial modification and passivisation. As a by-product of query expansion, other types of idiom variation captured include derivation, compounding, negation, distribution across multiple clauses as well as other unforeseen types of variation. The system was implemented on top of Elasticsearch, an open-source, distributed, scalable, real-time search engine. Flexible retrieval of idioms is supported by a combination of linguistic pre-processing of the search queries, their translation into a set of query clauses written in a query language called Query DSL, and analysis, an indexing process that involves tokenisation and normalisation. Our system outperformed the phrase search in terms of recall and outperformed the keyword search in terms of precision. Out of the three, our approach was found to provide the best balance between precision and recall. By providing a fast and easy way of finding idioms in large corpora, our approach can facilitate further developments in fields such as linguistics, language education and natural language processing.

Keywords: information retrieval; natural language processing; corpus linguistics; multi-word expressions; idioms

Citation: Hughes, C.; Filimonov, M.; Wray, A.; Spasić, I. Leaving No Stone Unturned: Flexible Retrieval of Idiomatic Expressions from a Large Text Corpus. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 263–283. https://doi.org/10.3390/make3010013

Academic Editor: Chris Biemann

Received: 2 February 2021
Accepted: 25 February 2021
Published: 3 March 2021

1. Introduction

An idiom is a sequence of words that constitutes a preferred way of expressing a given idea, even though it is not necessarily the most direct or obvious one. Many, though not all, idioms have a metaphorical meaning that in some way can still be linked to a literal one, e.g., 'fly in the face of', though sometimes changes in the language or culture strand the metaphorical interpretation without an easily retrievable literal counterpart, e.g., 'go to the wall', 'pig in a poke'. Old forms can become stranded inside fossilised expressions that could not now be generated using the active language forms, e.g., 'by and large', 'hi-fallutin', 'take umbrage'. In other cases, the most obvious-looking explanation for the origin may not be the correct one, e.g., 'give someone the cold shoulder', 'donkeys' years'. Yet there is a subset of idioms that are grammatical, transparent and logical, e.g., 'have a nice day' and 'don't do anything I wouldn't do'.

As far as we know, every language has idioms [1]. It also seems likely that at least some of the idioms in every language have peculiar characteristics like those just men-

tioned. This suggests that idioms exist for reasons relating to both cultural and cognitive behaviour. In cultural terms, their irregularity may play a part in social inclusion, exclusion and coherence, because their unpredictable forms and meanings mean that they are comprehensible only to those who have enough in-group experience to have encountered them and learned their meaning [2]. Cognitively, their irregularity may be a natural outcome of holistic processing, whereby their form and meaning are learnt and accepted without question, and thus not 'corrected' to keep up with the changing patterns of the language over time [3].

A key feature of idioms that is central to this paper is their peculiar mixture of fixedness and variability. In English, while some idioms have a completely unchangeable structure, e.g., 'by and large', 'lock stock and barrel', 'arms akimbo'; the majority do not. For instance, variability is generally inevitable if idioms contain a finite verb with an unspecified subject, because it needs to agree with the subject in person, number and tense and aspect (e.g., 'I/you/he/she/we/they (is/are/was/were/etc) weigh(s)/weighed/weighing up the options'). Verbs can also be passivised and modals can be added. In addition, many English idioms can accept optional adjectives and adverbs (e.g., 'a (very) tall story'; 'weather the (current) storm').

This intrinsic scope for variation within an idiom form, which varies across typologically different languages to create a range of additional challenges [1], constitutes a major logistical test for corpus linguists looking for occurrences of idioms. Which form(s) should they look for? Indeed, could they even anticipate all the different ways in which a given idiom might be realised? The answer to this second question is almost certainly no, for no sooner do we know an idiom than we start to play with it, in a process that Hanks [4] terms the exploitation of norms. In this process, we push at the boundaries of what others will recognise as an instance of the idiom, often for comic effect (e.g., 'joyful as a newt', a novel extension of 'pissed as a newt' used in a BBC radio drama [5].

Any computational approach to finding instances of an idiom will need something specific to look for: one or more words invariably associated with it. However, at the most extreme end of the variability continuum, there might be rather little such material available. Consider the idiom 'Is the Pope a Catholic?' If we treat this on its own, it is unproblematic, in that it does not vary. However, there is a set of alternative versions of this idiom—alternatives in the sense that they perform exactly the same semantic function—including 'Does a one-legged duck swim round in circles?' and 'Does a bear shit in the woods?' If these are all versions of the same idiom, there is no word that is always present, only an underlying frame consisting of an existential question (preferably somewhat comical or ridiculous) to which the answer is 'yes'. The corpus linguist will be hard pressed to find a way to search for instances of this idiom, even using tags. Wray [3] fortunately argues that there is no need to conceptualise these examples as one idiom, since it is much more difficult to create a new one than it is to vary an idiom. Rather, she proposes, each should be treated as a separate idiom, even though they are synonymous. Should reasons nevertheless be found to prefer viewing this set as variations on one idiom, this case may need to be considered the exception that tests the boundaries of the rule. Not being able to search for this very extreme case is a misfortune, not a tragedy.

Other major challenges include the inherent rarity of many idioms. In one of the first corpus investigations to tackle variability in idioms, Moon [6] found a striking absence of instances of many of her target idioms, despite making considerable efforts to find variations in the form. At a mere 18 million words, her corpus was, it is true, small by today's standards, but even very large corpora can provide remarkably few examples of idioms that are well-known enough to give the impression of being frequent.

Since variations in idioms are hard to search for, linguists often still use their intuition to judge how a given form may and may not change. The relationship between intuitions and evidence is something, however, that corpus linguistics has done much to question. Meanwhile, those researching their second language are at a significant disadvantage when intuition plays a major role, if they have had less exposure to rare examples, and are less

confident about judging what is possible. For these reasons, and because the challenge is there to be met, finding a way to more neutrally identify idioms computationally, across their broad range of possible realisations, is a research goal of some importance.

To that end, the main aim of this study was to create an idiom search engine that takes an idiom as input and finds all of its occurrences in a given corpus. The aim is to provide greater flexibility than simple string-matching techniques commonly used in existing search engines. By doing so, we aim to provide corpus linguists with a software tool that can help them investigate how idioms are used in discourse.

2. Related Work

This section reviews studies in natural language processing (NLP) that focus specifically on idioms. In particular, we focus solely on English idioms as the problems associated with processing of idioms in typologically different languages may vary considerably and any such comparisons are outside the scope of this study.

The majority of related studies focus on sense classification of potentially idiomatic expressions (PIEs), which are defined as multi-word expressions that cover both literal and non-literal uses of idiomatic expressions [7]. However, none of them provides a comprehensive algorithm for systematic retrieval of idiom mentions. More precisely, they largely focus on a subclass of idioms with relatively simple syntax (e.g., verb-noun constructions) and only address certain types of variation (e.g., substituting one pronoun by another). This review of related work aims to unravel this research gap and demonstrate the limited impact of idiom sense classification studies upon NLP due to their inability to systematically identify idiom mentions in free text. In that sense, our study is complementary to existing research in this area: when combined with existing sense classification approaches, idiom retrieval can be used to fully automate their recognition in free text. First, PIEs need to be retrieved and only then can their sense be classified. This positions our study as a catalyst in unlocking the use of idioms as features in NLP applications.

Interpreting PIEs requires automatic sense disambiguation between literal and idiomatic usage. For example, the PIE 'ring a bell' can be interpreted as 'sound vaguely familiar' (e.g., 'Well the name rings a bell.') or literally as 'ring a bell' (e.g., 'To enter you must ring a bell.'). The challenge of automatically differentiating between the two senses can be cast as a binary classification problem solved using machine learning approaches. For example, Cook et al. [8] proposed an unsupervised approach based on an assumption that idiomatic usages tend to occur in a small number of canonical forms for a given PIE, whereas the literal usages are less syntactically restricted and expressed in a greater variety of patterns. However, they only considered a subclass of idioms, verb-noun construction (VNC), which are formed from the combination of a verb and a noun in its direct object position for which such a strong assumption may well be valid. As the examples we will use later illustrate, this assumption does not generalise well to all idioms, which may exhibit a variety of deviations from their canonical forms.

Peng et al. describe a supervised approach for the binary classification of sentences into idiomatic and non-idiomatic [9]. Their approach is not confined to a particular linguistic construction such as VNC. This makes their approach more general than that of Cook et al. [8], although the supervised nature of their approach, which involved training on a small set of 600 idioms, may be another source of concern about generalisation. Our own approach does not involve training using specific idioms or targeting a narrow class of syntactic constructions. In Cook et al., the problem of potential overfitting is further accentuated by the choice of problem representation, which was based on the bag-of-words model with stopwords removed. Such representation strips the model of important information such as syntactic relations (including order) as well as relevant content. Our approach uses the concept of distance as means of syntactic approximation, which allowed us to preserve stopwords. Prepositions and conjunctions do play an important role in the formation of idioms with some consisting entirely of such words (e.g., 'out and about'), which are routinely deleted as stopwords. On the other hand, the focus on content words

does have advantages when the problem of binary sense classification is framed as that of identifying semantic (or lexical) outliers, i.e., observations that appear to be inconsistent with the discourse [10]. For example, when the PIE 'hot potato' co-occurs with other food-related words as the following example illustrates, then it is less likely to be used in its idiomatic sense: 'Put tomato pieces or whole cherry tomatoes among the drained, hot potatoes on serving plates and spoon sauce prettily over each serving.' In contrast, when the same expression co-occurs with unexpected words as the following example illustrates, then it is more likely to be used in its idiomatic sense: 'That's when you see the Colonel and his council playing pass-the-parcel with a hot potato.' This sort of distributional semantics approach has the additional advantage of not requiring class label information.

Liu and Hwa also confirmed that distributional semantics serves as a helpful heuristic for estimating the likelihood that a PIE is used literally [11]. Their approach was based on an assumption that a PIE's relationship with its context is more predictable when it is used literally than when it is used figuratively. To capture such predictability, they measured the semantic similarity between the context and the PIE's literal representation. Here, a PIE's literal usage is represented as a collection of the word embeddings of the PIE's constituent words and other words they co-occur with frequently. When a PIE is used in its literal sense, the words from its context are expected to be semantically closer to the literal usage representation. Cosine similarity is used to compare the words from the two collections (literal usage versus context) and determine average similarity. The higher the similarity, the higher the likelihood that the PIE is intended literally. Therefore, an appropriately chosen threshold can be used to differentiate between the literal and figurative meaning.

Sporleder and Li [12] proposed another unsupervised method for binary classification of PIEs based on their links with the overall cohesive structure of the discourse. The stronger the links, the higher the likelihood that a PIE is intended literally. Lexical cohesion is a property observed in texts where concepts discussed within individual sentences are typically related to concepts mentioned elsewhere [13]. Its practical usefulness depends on the choice of a method for computing semantic relatedness. Lexical knowledge bases such as WordNet [14] can be used to approximate semantic relatedness by using the shortest path in the semantic network between two concepts or by measuring the overlap of their glosses. Unfortunately, not all concepts are covered by WordNet and, for those that are, their connections are limited to thesaurus-like relationships. Alternatively, distributional approaches can be used to infer semantic relationships from the usage patterns in a large corpus. Sporleder and Li [12] used normalised Google distance (NGD), which computes relatedness based on the number of pages returned by the search engine for a query that combines two concepts [15]. A threshold applied to NGD values can be used to build a cohesion graph of the surrounding discourse. If removing the PIE's constituent words from such graph results in a higher connectivity calculated as the average edge weight, then the given PIE is assumed to be used non-literally. In their experiments, Sporleder and Li [12] focused on PIEs consisting of a verb followed by either a noun phrase or a prepositional phrase. Haagsma et al. [7] combined the two approaches proposed in [12] and [11]. They used the idea of lexical cohesion graphs but replaced NGD between two words by cosine similarity between the corresponding word embeddings.

Liu et al. [16] also explored the idea of lexical cohesion and word embeddings to learn to differentiate between literal and idiomatic meaning. Instead of comparing individual words from a PIE with its context, they compared the literal meaning of the PIE itself against the context. The literal meaning was calculated from the PIE's constituents using a tree-structured long short-term memory (LSTM) [17]. Rather than using a similarity measure to compare this meaning against the context, a neural model was used to detect the boundary between literal and idiomatic meaning. The surrounding context was represented as an LSTM-based function of its words. The binary classification of the intended meaning (literal versus idiomatic) was then obtained using a single layer multi-layer perceptron. Salton et al. [18] used a framework based on recurrent neural networks to encode a sentence into a distributed representation and then decode it to predict the neighbouring sentences.

Consequently, the context of a sentence is encoded without the need to access it explicitly. Based on that assumption, a supervised classifier of idiom usage can be trained on a set of distributed sentence representations without having to model long windows of context and without using methods to extract topic representations.

The related work described thus far focuses on automatic sense disambiguation between literal and idiomatic usage. Their main deficiency lies in the way in which they identify PIEs to disambiguate. When used in discourse, almost all idioms vary. The extent of these variations makes them difficult to retrieve from a large corpus using traditional brute-force approaches such as keyword or phrase search. More sophisticated ways of modelling variation in idioms computationally are required. One way is to model idioms using a local grammar approach. In an earlier study, we defined a set of lexico-semantic pattern-matching rules (represented by regular expressions) manually to automate subsequent recognition of idiom occurrences in text [19]. These rules were idiom specific, i.e., only idioms that were included explicitly in the model could be matched in text. To address the issues of generalisability and scalability, we later developed an approach to derive the pattern-matching rules automatically from idioms' canonical forms, i.e., their corresponding entries in a monolingual dictionary [20]. Flor and Beigman Klebanov [21] implemented a similar rule-based generation of pattern-matching rules, which were evaluated on a corpus of essays written by non-native English speakers. Although efficient algorithms for matching regular expressions have been in existence for a long time [22,23], widely used programming languages do not necessarily implement the most efficient solution available [24]. This study seeks to replace regular expressions by more efficient methods of finding idioms in a large corpus of text based on the state-of-the-art technology from information retrieval. By doing so, this study aims to fill the gap that is prominent in all related work presented: finding idioms in text is a prerequisite for any automated reasoning about idiom usage. None of the existing approaches offers a comprehensive solution to this problem that is both generalisable and scalable. This study aims to address this very problem. Our own approach does not include disambiguation, but it can easily be extended to support such functionality by combining idiom retrieval with any of the classification approaches described earlier. However, this particular problem is beyond the scope of the current study.

3. Methods

3.1. Technology

In this section, we provide details about the chosen technology in a quest to justify its choice as well as facilitate a reader in understanding the details of implementation specific to our application.

3.1.1. Elasticsearch

Elasticsearch is an open-source, distributed, scalable, real-time search engine [25]. It is built on top of Apache Lucene, an advanced, high-performance, full-text search engine library [26]. Elasticsearch uses Lucene internally for indexing and searching but hides its inner complexity behind a simple-to-use RESTful API, which allows programming languages to communicate with Elasticsearch using a web client.

Elasticsearch stores documents formatted in JavaScript Object Notation (JSON), each representing a hierarchy of fields. By default, every field is indexed, and, hence, searchable. An inverted index associates each word with a list of documents in which it appears. Such data structure greatly accelerates the process of full-text searching. In addition, an index in Elasticsearch is implemented as a collection of shards. In other words, an index is a logical namespace that points to one or more physical shards. Effectively, documents are stored and indexed in shards, which are allocated to nodes (servers), which work together in a cluster to increase the availability, scalability and reliability of the search engine. This is how Elasticsearch distributes data. Elasticsearch migrates shards between nodes to maintain the balance within a cluster as it grows or shrinks. Applications do not need to

communicate to shards directly. Instead, they communicate with an index, which simplifies the development of applications that require scalable full-text searching.

3.1.2. Indexing

To improve the "searchability" of documents, indexing in Elasticsearch uses analysis, a process that involves character filtering, tokenisation and normalisation (or token filtering). Character filters are used to tidy up the text, e.g., by stripping out mark up, converting special characters, etc. A tokeniser segments the filtered text into tokens, each being a sequence of characters grouped together as a unit for further processing. Token filters are applied to individual tokens to modify them (e.g., lowercase), remove them (e.g., stopwords) or associate them with other tokens (e.g., synonyms). Such functionalities are integrated into a single package called an analyser, which can be configured using the syntax given in Figure 1. The full-text fields are analysed during indexing. For a query to match indexed content, it needs to undergo the same analysis. When a full-text field is queried, the query automatically applies the analyser associated with the field to produce a list of search terms compatible with the inverted index.

```
{
   "settings": {
      "analysis": {
         "char_filter":   ... custom character filter(s) ...,
         "tokenizer":     ... custom tokeniser(s) ...,
         "filter":        ... custom token filter(s) ...,
         "analyzer":      ... custom analyser(s) ...
      }
   }
}
```

Figure 1. Defining an analyser in Elasticsearch.

3.1.3. Querying

Elasticsearch provides its own query language called Query DSL (domain-specific language) whose syntax is also based on JSON. Query clauses are building blocks that can be combined to create complex queries. A query clause typically follows the structure given in Figure 2.

```
{
   query_name: {
      field_name: {
         argument: value,
         argument: value,
         ...
      }
   }
```

Figure 2. The structure of a query clause in Query DSL.

A leaf clause is used to match one or more fields against a query string. A compound clause combines other query clauses in a hierarchical fashion. Each query clause generates a relevance score for each document. Its calculation depends on the type of query clause, e.g., fuzzy query calculates how similar two matched strings are, a terms query incorporates the percentage of search terms that were matched successfully, etc. In general, relevance score is based on the similarity between the contents of a full-text field and a full-text query string. The relative weight of any query clause can be controlled by specifying a boost value. Ultimately, the results of full-text searching are ranked by relevance score, i.e., the extent to which a document matches the query. When a query is complex, it can become difficult to understand how the relevance score was calculated. Therefore, Elasticsearch provides an option to attach an explanation of the specific score value to every search result. In addition, the relevant snippets of text can be highlighted so that the user can see how a retrieved document matches the query.

Standard full-text search treats each document as a bag of words. The match query simply determines whether a document contains the search terms and does not consider the relationships among them, e.g., order or distance. On the other hand, the match_phrase query matches documents that contain all search terms but only when they maintain the same positions relative to one another. For example, the query given in Figure 3 will only match documents that contain the three given words, 'scream', 'blue' and *'murder'*, when they occur in the exact same order.

```
{
   "query": {
      "match_phrase": {
         "document": "scream blue murder"
      }
   }
}
```

Figure 3. An example of a match_phrase query.

Positions (within a document) can be stored in an inverted index to be utilised at run time by position-aware queries such as the match_phrase one. Phrase matching may be too rigid—ignoring many phrases with only minor variations. The slop parameter allows for some flexibility in phrase matching by allowing the search terms to be spaced apart within a document while still considering it a match (see Figure 4 for an example). More precisely, the slop parameter prescribes the number of word position changes allowed within a document for a phrase to match exactly. This means that with a sufficiently large slop value, words do not necessarily need to appear in the same order. However, a higher relevance score is given to documents in which the search terms are found closer together.

Although proximity queries introduce a degree of flexibility into phrase searching, the fact that they require all search terms to be found within a document may still be too rigid. Rather than using proximity matching (i.e., phrase matching with a slop) as an absolute requirement, we can make it a desirable one within a bool query, where it can be combined with other queries under the must, must_not and should parameters for additional flexibility (see Figure 5 for an example). For example, an exact match query with the minimum_should_match parameter can be used as a must clause to constrain the search space followed by more sophisticated queries targeting different aspects of document relevance as should clauses. Every query clause that matches will increase the relevance of the corresponding documents.

```
{
   "query": {
      "match_phrase": {
         "document": {
            "query": "scream murder",
            "slop": 1
         }
      }
   }
}
```

Figure 4. An example of a proximity query.

```
{
   "query": {
      "bool": {
         "must": {
            "match": {
               "document": {
                  "query": "scream blue murder",
                  "minimum_should_match": "75%"
               }
            }
         },
         "should": {
            "match_phrase": {
               "document": {
                  "query": "scream blue murder",
                  "slop": 5
               }
            }
         }
      }
   }
}
```

Figure 5. An example of a bool query.

3.2. Implementation

3.2.1. Data

The British National Corpus (BNC) is a large corpus of British English texts from 1960 onwards [27]. It is composed of text samples of both spoken and written language, each no longer than 45K words and chosen to be as varied as possible to achieve a wide coverage of subject fields, registers and genres. Though not the largest corpus, the advantage of using the BNC in linguistics research is the fact that it is balanced, thus allowing for extrapolation of properties that apply to the language in general. The XML Edition of the BNC contains 4049 texts comprising almost 100M orthographic words. The BNC XML edition is marked up in XML and encoded in Unicode [28], which can be easily converted into other formats as necessary using XML-aware processing tools. The text content is

tokenised and tagged with part of speech (POS), resulting in tokens being represented as separate XML elements and grouped hierarchically into sentences and paragraphs also represented as XML elements (see Figure 6). XML can easily be converted into JSON format, which can then be imported into Elasticsearch.

```
<bncDoc xml:id="CAD">
...
  <s n="2710">
    <c c5="PUQ">'</c>
    <w c5="NN1" hw="rock'n'roll" pos="SUBST">Rock'n'roll</w>
    <w c5="VVZ" hw="open" pos="VERB">opens</w>
    <w c5="AVP" hw="up" pos="ADV">up</w>
    <w c5="DT0" hw="these" pos="ADJ">these</w>
    <w c5="NN2" hw="floodgate" pos="SUBST">floodgates</w>
    <c c5="PUN">,</c>
    <c c5="PUQ">'</c>
    <w c5="PNP" hw="she" po="PRON">she</w>
    <w c5="VVZ" hw="insist" pos="VERB">insists</w>
    <c c5="PUN">.</c>
  </s>
...
</bncDoc>
```

Figure 6. A sample sentence from the BNC formatted in XML.

Alternatively, a character filter can be applied during indexing to strip off the XML mark up and only import raw text. Neither is entirely appropriate for our application—the retrieval of idioms. In theory, idioms such 'bury the hatchet' and 'dig up the hatchet' can have their lexical content distributed across sentences as the following example taken from a blog [29] illustrates: 'Look I know they buried the hatchet. Then Uma dug it up. Then she reburied it. But I think she buried it in a shallow grave.'

However, idioms generally tend to be encompassed by single sentences. Therefore, sentences were chosen to represent the basic units of retrieval in our application. Given that sentences have already been marked up in the BNC XML edition, we converted the content of this particular XML element into the corresponding full-text field in JSON (see Figure 7).

```
{
   "sentence": {
      "id": "CAD 2710",
      "text": "'Rock'n'roll opens up these floodgates,' she insists."
   }
}
```

Figure 7. A sample sentence from the BNC formatted in JSON.

The Natural Language Toolkit (NLTK) corpus package defines a collection of corpus reader classes [30], which can be used to access the content of various corpora including the BNC. Each corpus reader class is specialised to parse a corpus-specific format. The BNC corpus reader was used to extract sentences from the XML edition. As Figure 6 illustrates, the content is tokenised with each word represented by a separate XML element. Each XML-formatted sentence was then detokenised using the NLTK TreebankWordDetokenizer class to remove the mark up but also to restore the raw text compliant with English writing norms, which prescribe conventional ways of using punctuation and white spaces. A total of 6,026,276 sentences were extracted from the BNC XML edition and imported into Elasticsearch database in the JSON format given above (see Figure 7).

3.2.2. Indexing

A custom analyser was created to index sentences in a way that supports sensitivity and transparency of retrieving various forms of idioms (see Figure 8). In Elasticsearch, the standard analyser, which is the default for full-text fields, does not remove stopwords. Prepositions, conjunctions and determiners, which are typically removed as stopwords, play an important role in the phrasal stability of idioms. For example, removing them from idioms such as 'in the teeth of', 'be up in arms' or 'on the up and up' would also remove the overall meaning. Therefore, although stopwords can easily be added to a custom analyser, we retained the default position not to remove them. Our custom analyser starts by expanding enclitics, removing apostrophes and lowercasing alphabetic characters.

To understand the role of other functionalities that were integrated into the analyser, we first need to review the ways in which idioms may vary. The difficulty associated with searching for idioms in a big corpus is the fact that idioms are heterogeneous in terms of their transformational capacity [31]. While some idioms allow virtually no variation without the loss of the idiomatic sense, others may exhibit extensive variation [6,32,33]. In this study, we focus on the most common types of idiom variation, namely inflection, open slots, adjectival or adverbial modification and passivisation [34]. Some of these phenomena can be addressed at indexing time, whereas others need to be resolved at querying time.

In terms of inflection, verbs can be used in different tenses, nouns can be used in singular or plural, whereas adjectives can be inflected to express different degrees of comparison. For example, the verb in the idiom 'jump the gun' was used in the present perfect tense in the following example: 'But I have jumped the gun.' Similarly, the noun in the idiom 'cross to bear' is used in plural the following example: 'I know you haven't it's just one of these crosses you've got to bear haven't you?'

To support retrieval of inflected forms, stemming both the canonical form of an idiom (i.e., the search query) and the sentences to be matched (i.e., indexed document) would effectively neutralise the inflection. In Elasticsearch, stemming is handled by stemmer token filters, which can be either algorithmic stemmers (i.e., based on transformational rules) or dictionary stemmers (i.e., based on dictionary lookup). In practice, algorithmic stemmers usually outperform dictionary stemmers in terms of coverage (e.g., stemming of unknown words) and efficiency (i.e., time and memory requirements). For these reasons, the Porter stemmer [35] was selected as a token filter and integrated into our custom analyser. As a result, a stem rather than the original surface form is stored in the index. For a search query to match indexed content, it needs to undergo stemming as well. When a full-text field is queried, Elasticsearch automatically launches the analyser associated with the field. In our case, this means that a search query will always get stemmed to produce a list of search terms compatible with the inverted index.

```
{
   "analyzer": {
      "type": "custom",
      "char_filter": [ "enclitics_01", "enclitics_02", ... ],
      "tokenizer": "standard",
      "filter": [
         "apostrophe",
         "lowercase",
         "synonyms",
         "stemmer_exceptions",
         "porter_stemmer"
      ]
   },
   "char_filter": {
      "enclitics_01": {
         "type": "pattern_replace",
         "pattern": "aren't",
         "replacement": "are not",
         "flags": "CASE_INSENSITIVE"
      }, ---
   },
   "filter": {
      "synonyms": {
         "type": "synonym",
         "synonyms_path": "txtfiles/synonyms.txt"
      },
      "stemmer_exceptions": {
         "type": "stemmer_override",
         "rules_path": "txtfiles/exceptions.txt"
      },
      "porter_stemmer": {
         "type": "stemmer",
         "name": "english"
      }
   }
}
```

Figure 8. A custom analyser for indexing idioms.

By and large, stemming will resolve the problem of matching inflected word forms. However, one shortcoming of algorithmic stemmers is that they do not cope well with irregular words, so the problem of irregular words persists. For example, idioms such as 'swim against the tide', 'wolf in sheep's clothing' and 'stand in good stead' will not match their mentions in the following examples:

> 'He had swum against the tide.'
>
> 'But now we know they're just wolves in sheep's clothing.'
>
> 'A polite and considered approach, avoiding outright confrontation, will stand you in better stead.'

Irregular words represent a case where dictionary stemmers perform better. Rather than reverting to dictionary stemmers, we can use a small dictionary for irregular words only. Elasticsearch can match different tokens by using the synonym token filter. Although, as the name suggests, this functionality is used primarily to conflate synonyms, i.e., different words with the same meaning (e.g., 'jump', 'leap' and 'hop'), it can just as well be used to match different forms of irregular words. To patch the stemming of irregular words, we complemented the stemmer token filter with the synonym token filter designed specifically to model inflection of irregular words.

In addition to inflection, some other types of idiom variation are simple enough to be modelled by the synonym token filter. For example, idioms often feature a pronoun used as an anaphor, which can vary in terms of number, gender and case to match that of the antecedent as the following instance of the idiom 'butterflies in one's stomach' illustrates: 'The morning she saw him she suddenly felt butterflies in her stomach.'

By indexing pronouns as synonyms, they can be flexibly replaced by one another at querying time. In addition to pronouns, some content words can sometimes be replaced while still preserving the idiomatic sense, e.g., 'take a load/weight off someone's mind' or 'swim against the stream/tide'. Although, individual words are not necessarily synonyms themselves, the overall figurative sense remains synonymous. Given that this variation is confined to the token level, it too can be modelled by the synonym token filter. To identify tokens that can replace one another in idioms, we parsed a list of idiom definitions acquired from Learn English Today, a free website for learners of English [36]. Equivalent tokens were identified using the slash character as indicated in the examples above, e.g., load/weight and stream/tide.

3.2.3. Querying

Modification

The components of some idioms are modifiable, e.g., nouns and verbs can be modified by adjectives and adverbs, respectively. The following example of the idiom 'grasp at straws' contains both types of modification: 'Vologsky grasped desperately at the floating straw.' A simple phrase search query would fail to retrieve idioms that contain inserted modifiers. As we discussed earlier, Elasticsearch supports phrase matching with a slop, which allows a certain number of words to be inserted between the matching words from the search phrase. A fixed slop value may be too generous for short idioms, thus reducing the precision. On the other hand, it may be too strict for long idioms, thus reducing the recall. The slop value should be set so that consistent performance is achieved across all idioms regardless of their length. More precisely, it is the number of modifiable elements that should be used to determine the slop rather than the raw length. We estimated the slop value by counting the number of relevant parts of speech. Prior to launching a search, NLTK is used to tag the search phrase (i.e., the canonical form of the idiom) provided by the user. POS tags are then used to count the total number of nouns and verbs as potentially modifiable components. The dynamically calculated slop and the input idiom are combined into a phrase search query.

Open Slots

Many idioms contain pronouns as open slots that can be substituted by other phrases. For example, in the idiom 'keep someone at arm's length' the open slot, which is indicated by an indefinite pronoun, is substituted by a noun phrase in the following example: 'They preferred to persist in Piłsudski's strategy of keeping both Germans and Russians at arm's length.' The following example illustrates even more complex substitution: 'They may prefer the limited protection provided by keeping each other (and their own needs) at arm's length by replaying the old scenarios.' The most basic substitutions are addressed at indexing time by using the synonym token filer, which allows one pronoun to be replaced by another pronoun. However, as the two examples demonstrate, pronoun substitutions are not limited to a closed vocabulary and, hence, require a pattern-matching approach at querying time. To account for open slots, we can identify pronouns in the original search phrase, which has already been tagged with POS. Each pronoun is then removed from the query and the previously calculated slop value is increased to allow room for its substitution by more complex phrases.

Passivisation

In addition to inflection, verbs in idioms may vary in terms of their voice too. Passive voice allows the object of an otherwise active sentence to become the subject of a passive sentence. In this process, the order between the verb and its object gets reversed. For example, compare an active form of the idiom 'open the floodgates' 'The case could open the floodgates for thousands of similar claims worldwide.' to a passive one 'And with Wright gone, the floodgates were opened.' To account for the passivisation of idioms, POS tags can be used to identify non-auxiliary verbs at the beginning of an idiom and produce an additional search query for its passive form by repositioning the verb to the end of the idiom. A slop is used to accommodate the auxiliary verb in front of the past participle of the given verb (e.g., 'were' in the last example) and its potential modification. If the idiom also contains a pronoun, this suggests a passivised version of the idiom may contain an open slot. Therefore, an open slot in the passivised query is processed as described earlier by replacing the pronoun by a wildcard and increasing the slop further. Table 1 provides a summary of querying approaches taken to address specific types of idiom variation together with a run-through example.

Table 1. Summary of query generation.

ID	Variation	Problem	Solution	POS	Example Query
1	modification	insertion	slop	noun + verb	{"query": "call someone's bluff", "slop": 4}
2	open slot	replacement	wildcard + slop	pronoun	{"query": "call * bluff", "slop": 5}
3	passivisation with modification	reordering + insertion	reordering + slop	verb	{"query": "someone's bluff * call", "slop": 5}
4	passivisation with an open slot	reordering + replacement	reordering + wildcard + slop	verb + pronoun	{"query": "* bluff * call", "slop": 6}

Up to four search queries are generated for a given idiom. Each query is represented by a separate clause. A single bool query is then used to combine all queries as should clauses (see Figure 9). If a should clause matches a document, then it increases its overall relevance score, otherwise it has no effect. A similarity model used to score matching documents is the default—BM25 [37], which is based on the probabilistic relevance model and is considered to be a state-of-the-art ranking function. No bespoke boosting was applied to different clauses of the query. However, documents matching the first clause of the query will score highest.

```
{
   "query": {
      "bool": {
         "should": [
            {"match_phrase": {"sentence": query1}},
            {"match_phrase": {"sentence": query2}},
            {"match_phrase": {"sentence": query3}},
            {"match_phrase": {"sentence": query4}}
         ]
      }
   }
}
```

Figure 9. A bool query combining queries given in Table 1.

Although originally developed to address specific types of idiom variation outlined in Table 1, the combined query is flexible enough to handle other types of variation as the examples provided in Table 2 illustrate. The stemming filter applied during indexing will neutralise not only inflection but derivation as well. The first example contains a nominalised instance of an idiom as does the second, which also combines it with compounding. Modification is not necessarily restricted to adjectives and adverbs. Negated verbs can also be retrieved (see examples 3 and 5). More complex prepositional phrases can be used to modify nouns within idioms (see example 4). The components of some idioms may be distributed over multiple clauses (see example 5). Some unforeseen variations such as emulation of stuttering have been captured as well (see example 6).

Table 2. Other types of idioms variation covered.

ID	Variation	Idiom	Example
1	derivation	bury the hatchet	These occasions are marked by much conviviality and the temporary burying of hatchets.
2	compounding	grease someone's palm	Palm-greasing for just about anything from entry to a favoured school to obtaining a bank loan has been considered a fact of life.
3	negation	grease someone's palm	The gondoliers threatened to go on strike and all the floodlights on the night of the show were mysteriously switched off because someone hadn't had their palm greased.
4	modification by a prepositional phrase	open the floodgates	The floodgates to total permissiveness were opened and a society in which "the permissive intellectual's anything goes" was created.
5	distribution over multiple clauses	born with a silver spoon in one's mouth	She was born, if not with a silver spoon in her mouth, then certainly not one with any chicken soup on it.
6	orthographic simulation of stutter	head over heels	H-have to admit it, old thing, I'm h-head over h-heels in love with you.

3.2.4. Search Results

As explained before, each retrieved document (in our case is an individual sentence) will receive a relevance score, which reflects how well it matches a query. The relevance score is represented by a positive floating-point number with higher values reflecting higher relevance. As noted earlier, a bool query combines multiple should clauses, so the overall score will aggregate their individual scores, which can make it difficult for a user to understand how it was calculated. Elasticsearch can produce an explanation of the score for every search result by setting the explain parameter to true. We provide the relevance score with every search result as a clickable link, which brings about an explanation on demand.

Elasticsearch also allows for the matching snippets of text to be highlighted in the search results. We added a JSON element (see Figure 10) to the search query that prescribes that the matching tokens should be automatically enclosed by the chosen XML tags. These tags are then used to highlight the matching text in HTML for the user's to visual inspection but also to mark idioms up in a downloadable text file for the user to use offline with other tools of their choice.

```
{
   "highlight": {
      "fields": {
         "sentence": {
            "number_of_fragments": 1,
            "fragment_size": 1000,
            "pre_tags": "<idiom>",
            "post_tags": "<\/idiom>"
         }
      }
   }
}
```

Figure 10. Highlighting matching snippets in the search results.

4. Results

4.1. Baseline

Earlier in this paper, we argued that the extent of variation in the forms of idioms makes them difficult to retrieve from a large corpus using traditional brute-force approaches such as keyword or phrase search, and hence we proposed a sophisticated search strategy that addresses such variation automatically. To test the efficacy of the new search strategy it therefore makes sense to compare it to the two search options that we have claimed are inadequate. The first baseline is the keyword search, which matches the search terms using their stems and ignoring their order and distance within the matching documents. The second baseline is the phrase search, which matches the search query as an exact phrase not allowing for any variation in word order or their morphology. Both baseline approaches are commonly supported by search engines, so they represent strategies that users looking for idioms in large corpora would be most likely to employ. Therefore, they represent a standard benchmark to gauge the effectiveness of search performance.

4.2. Measures

The ideal aim of information retrieval is to collect all and only those documents that are relevant to the search. Any shortfall in this outcome is indicative of limitations in the search design. A means is needed for calculating the efficacy of the search. To this end, any document retrieved by a system can be classified either as a true positive (TP) if it

is relevant to the user's information need expressed by the query or as a false positive (FP) otherwise. Conversely, any relevant document from the given collection that is not retrieved by the system constitutes a false negative (FN). Using the total numbers of TPs, FPs and FNs, precision (P) and recall (R) are calculated simply as the following ratios and combined into the F score as their harmonic mean:

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \qquad \mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad \mathrm{F} = \frac{2 \cdot \mathrm{P} \cdot \mathrm{R}}{\mathrm{P} + \mathrm{R}} \tag{1}$$

Precision measures the proportion of correctly retrieved documents, while *recall* measures the proportion of relevant documents that are retrieved by the system. To calculate precision, the retrieved documents need to be inspected manually in order to differentiate between TPs and FPs.

To calculate recall, the whole document collection needs to be inspected manually in order to differentiate between TPs and FNs. When the document collection is as large as the BNC, measuring recall becomes highly impractical. However, the total number of relevant documents, i.e., the sum of TPs and FNs, is independent of the search strategy used. In other words, the recall denominator remains constant regardless of the actual search. Therefore, when comparing different search strategies in terms of recall, it suffices to compare just the numerators, i.e., the numbers of TPs retrieved. It is still desirable to normalise the number of TPs on a scale from 0 and 1. The relative recall achieves this by dividing the number of TPs by the total number of relevant documents retrieved by any of the considered search strategies [38].

4.3. Testbed

The PIE Corpus was developed to support the automatic detection of potentially idiomatic expressions [7]. In addition to figurative uses, PIEs also encompass literal ones such as 'The sharp blade gave me a close shave in one go, but I also cut myself twice' for the idiom 'close shave' whose figurative sense is 'a situation that was nearly an accident'. The corpus uses a pre-defined set of 591 PIEs and consists of 2239 sentences extracted from a set of 23 documents from the BNC. Each sentence may contain a PIE and is annotated for its presence. If it does, the given PIE is annotated according to its sense: literal or figurative. Unfortunately, the number of sentences per idiom was too small to evaluate the recall. Moreover, a closer inspection of the idiom types revealed that the vast majority seemed to be fixed phrases, which would not sufficiently stretch the system.

Therefore, we opted to randomly select 100 out of around 3000 idioms available from an educational web site—Learn English Today [36]. Nonetheless, we were able to re-purpose the original annotation guidelines from the PIE Corpus. Given a test idiom, every sentence retrieved automatically was annotated manually for whether it contained the given idiom regardless of its sense. If it did, the sentence was further annotated with the sense the idiom was used in, *idiomatic*, *literal*, *unclear* and *other*. The sense was deemed *unclear* when the sentence itself did not provide sufficient context to differentiate between *idiomatic* and *literal* sense. The sense was annotated as *other* when it was certainly neither *idiomatic* nor *literal*.

For each idiom from the testbed, we ran three searches: (1) the flexible search using the system described in this paper, (2) the keyword search and (3) the phrase search. These searches retrieved three ranked lists of sentences from the BNC. To make the annotation workload manageable, the lists were truncated to 100 top-ranked sentences. We effectively measured precision and recall at k, where $k = 100$. This approach is commonly used to evaluate ranked search results, where users are normally expected to start at the top working their way down the ranked list of retrieved documents. However, from the linguistic point of view, the most interesting examples of idioms, i.e., those that differ the most from their canonical forms as measured by the slop parameter of phrase search, would be ranked lower. Our choice of the parameter k allowed us to cut off the long tail of search results while preserving the expected number of idioms based on their distribution.

According to [6], we can expect 88%, 11% and 1% of idioms to occur <1, 1–5 and 5–50 times per million words respectively. If we project the expected frequencies to 100 M words in the BNC, we can see that we can expect to find less than 100 mentions for the vast majority of idioms. Indeed, the average number of sentences retrieved for the phrase, keyword and flexible search was 13.57, 29.52 and 49.89 respectively. The truncation had to be performed only in 2, 6 and 8 cases respectively.

The three lists of retrieved sentences, one for each search method, were merged, duplicates removed to reduce the manual annotation effort and the order randomised to reduce the annotation bias. The data were annotated by a single annotator.

For quality control purposes, a sample of data were annotated by the second annotator. For each idiom, up to two sentences, preferably one with *idiomatic* and one with another (*literal*, *unclear* or *other*) sense, were selected randomly. The confusion matrix shown in Table 3 compares the labels provided by the two annotators. Their agreement was measured using Cohen's kappa coefficient [39], which is calculated according to the following formula:

$$\kappa = 1 - \frac{1 - p_o}{1 - p_e} \tag{2}$$

where p_o is the observed agreement (i.e., the proportion of items on which both annotators agree) and p_e is the expected chance agreement calculated under the assumption that annotators act independently of each other and that random assignment of labels is governed by their distribution. At κ = 0.9043 with a standard error of 0.0294 [40] and a 0.95 confidence interval of 0.8468 to 0.9618, the inter-annotator agreement was found to be very good using the following scale [41]: 0–0.20 (poor), 0.21–0.40 (fair), 0.41–0.60 (moderate), 0.61–0.80 (good), 0.81–1.00 (very good).

Table 3. Confusion matrix on the double-annotated subset.

Label	Idiomatic	Unclear	Literal	Other	Total
Idiomatic	122	0	2	1	125
Unclear	3	4	0	0	7
Literal	0	0	19	0	19
Other	3	0	1	43	47
Total	128	4	22	44	198

The final testbed consists of 100 idioms and up to 100 sentences per idiom, where each sentence was annotated with the sense the given idiom was used in: idiomatic, literal, unclear or other. Note that our system only performs retrieval of PIEs without any attempt to disambiguate them. In that respect, we only need to differentiate between PIEs (positive instances) and non-PIEs (negative instances). That means that, for the purpose of this study, we aggregated the idiomatic, literal and unclear labels to represent PIEs and used the other label to represent non-PIEs. The main reason for performing annotation that was finer-grained than necessary was to create a dataset that could also support future studies on sense disambiguation. Knowing that manual annotation is costly and time-consuming endeavour, it is worthwhile to maximise its utility for further research. The testbed is publicly available at https://github.com/ispasic/idiometry/tree/master/data (accessed on 1 January 2021).

4.4. Results

Once the merged search results were annotated, the labels were distributed across the original three lists, one for each search method. Within each list, all sentences retrieved for a given idiom were assorted into true and false positives. A sentence was regarded to be a TP is it contained the corresponding PIE (i.e., it was previously labelled idiomatic, literal or unclear), otherwise it was considered an FP (i.e., its label was other). Precision, relative recall and their harmonic mean were then calculated. Table 4 provides these values for a sample of 10 idioms. Precision, recall and F score were micro- and macro-averaged across all 100 idioms for each search method (see Table 5).

Table 4. A sample of flexible search results.

Idiom	TP	FP	P	R	F
bitten by the bug	11	0	100.00%	100.00%	100.00%
blot one's copy-book	5	0	100.00%	100.00%	100.00%
come up in the world	8	2	80.00%	22.86%	35.56%
lose one's marbles	16	2	88.89%	100.00%	94.12%
get the better of someone	98	2	98.00%	51.31%	67.35%
give the all clear	39	1	97.50%	66.10%	78.79%
go out of one's way	1	5	16.67%	12.50%	14.29%
risk life and limb	24	1	96.00%	96.00%	96.00%
sink or swim	31	0	100.00%	96.88%	98.41%
turn the tide	98	2	98.00%	73.13%	83.76%

Table 5. Evaluation results.

	Micro-Average			Macro-Average		
Method	**P**	**R**	**F**	**P**	**R**	**F**
Phrase search	99.92%	31.20%	47.55%	81.98%	33.35%	47.41%
Flexible search	95.33%	82.79%	88.62%	95.28%	85.92%	90.36%
Keyword search	73.06%	44.63%	55.42%	72.54%	49.44%	58.80%

As Table 4 shows, there was some variation in retrieval performance across the idioms. The system will overgeneralise in cases where the words from idiom tend to co-occur with one another as the following FPs for idiom 'go out of one's way' illustrate:

> 'Work out a rough way of what you're going to give for.'
>
> 'Some may go to out-of-the-way places to sniff which can add to the dangers.'
>
> 'We went out separate ways, nearly all of us to be affected in one way or another by the Bodyline tour.'

However, a closer inspection of the search results showed that such behaviour was an exception rather than a rule. Indeed, the average precision of 95% (see Table 5) confirmed the fact that fewer FPs were typically retrieved. This may come at the cost of recall, which was found to be less stable (see Table 4). Nonetheless, average recall was over 80%, which was found to be much higher than either of the baselines (see Table 5). Somewhat surprisingly, our method outperformed the keyword search in terms of recall. This is because the standard English analyser in Elasticsearch does not remove the stopwords. Had the stopwords been removed, then the recall of keyword search would have been much higher. However, by loosening up the search conditions the precision of keyword search would have fallen even more dramatically. As it stands, it was already poorer by more than

20 percentage points. When precision and recall were combined, the performance of our method was found to be superior improving the F score by at least 30 percentage points.

5. Conclusions

We have described a system for the large-scale retrieval of idiom mentions. It takes a (presumably) canonical form of an idiom and expands it automatically to account for the most common types of idiom variation including inflection, open slots, adjectival or adverbial modification and passivisation, but also derivation, compounding, negation, distribution over multiple clauses and other unforeseen types of variation. The system provides a selection of almost 3000 idioms that a user can use to explore the search functionality, which is by no means limited to this list. In other words, a user can search for idioms of their own choice.

We envisage that the prospective userbase will consist primarily of linguists and teachers or learners of English who can use the system to gain new insights into idiom usage. Linguists can easily access the data that can be used to challenge (or verify) existing theories on idioms, including their variation, compositionality, perception, etc. Teachers can retrieve real-world examples of idiom usage that can be used to test English proficiency. Like any other words and phrases, idioms need to be learnt in context in order to prevent systematic forgetting [42]. Our system can be used to extend well beyond the examples of idiom usage provided in traditional textbooks. The links to full documents in the BNC can be used to set these examples into wider context.

Although the online system only provides access to the BNC, the search method itself is applicable to any corpus. The codebase is shared under open access license, which allows other corpora to be imported into its local installations. Moreover, the search method can be easily integrated into other NLP applications to mark up PIEs, which can then be used to support the recognition of multi-word expressions and classification of their usage (literal versus figurative). Two particular areas of NLP that can benefit from this are sentiment analysis and machine translation.

Deep learning approaches were found to learn some features repeatedly across multiple networks, but rare features such as idioms are not always learnt [43]. Yet, rare features tend to improve text classification accuracy [44,45]. Indeed, our previous study on sentiment analysis identified idioms as very predictive but comparatively rare features [19]. Our approach allows sentiment analysis to use idioms as features, which would otherwise be difficult to learn automatically using machine learning approaches.

Similarly, despite the success for neural machine translation in generating continually improving translations, translation of idioms remains an open challenge [46]. The first step towards addressing this challenge is the creation of data sets for learning and evaluating idiom translation. Our approach allows for large-scale identification of idioms in English as the source language. By aligning them with translations in the target language, training data for idiom translation can be created automatically.

Author Contributions: Conceived the idea and designed the system, I.S.; implemented the core functionality of the system, C.H.; adapted the system for online use, M.F.; helped refine user requirements from a perspective of a linguist, A.W.; drafted the manuscript, I.S., C.H. and A.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data cited herein have been extracted from the British National Corpus, distributed by the University of Oxford on behalf of the BNC Consortium. All rights in the texts cited are reserved.

Acknowledgments: We are thankful to Kathleen Beke, the creator of the web site Learn English Today, for kindly letting us use their data.

Conflicts of Interest: The authors declare no conflict of interest.

Code Availability: The demo of the system is available at https://datainnovation.cardiff.ac.uk/idiometry (accessed on 1 January 2021). The code is shared at https://github.com/ispasic/idiometry (accessed on 1 January 2021) under GPL v3 license. The annotated dataset used to evaluate the system is also provided with the code.

References

1. Buerki, A. (How) is formulaic language universal? Insights from Korean, German and English. In *Formulaic Language and New Data: Theoretical and Methodological Implications, Formulaic Language*; Piirainen, E., Filatkina, N., Stumpf, S., Pfeiffer, C., Eds.; De Gruyter: Berlin, Germany, 2020; Volume 2, pp. 103–134.
2. Wray, A.; Grace, G.W. The consequences of talking to strangers: Evolutionary corollaries of socio-cultural influences on linguistic form. *Lingua* **2007**, *117*, 543–578. [CrossRef]
3. Wray, A. *Formulaic Language and the Lexicon*; Cambridge University Press: Cambridge, UK, 2002.
4. Hanks, P. *Lexical Analysis: Norms and Exploitations*; MIT Press: Cambridge, MA, USA, 2013.
5. BBC Radio 4. Spoilers for May 25th–28th 2020. Available online: https://www.facebook.com/notes/archers-appreciation/spoilers-for-may-25th28th-2020/851327765348107/ (accessed on 1 January 2021).
6. Moon, R. *Fixed Expressions and Idioms in English: A Corpus-Based Approach*; OUP Oxford: Oxford, UK, 1998; p. 356.
7. Haagsma, H.; Nissim, M.; Bos, J. The other side of the coin: Unsupervised disambiguation of potentially idiomatic expressions by contrasting senses. In Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions, Santa Fe, NM, USA, 25–26 August 2018; pp. 178–184.
8. Cook, P.; Fazly, A.; Stevenson, S. Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In Proceedings of the Workshop on A Broader Perspective on Multiword Expressions, Prague, Czech Republic, 28 June 2007; pp. 41–48.
9. Peng, J.; Feldman, A.; Street, L. Computing linear discriminants for idiomatic sentence detection. *Res. Comput. Sci.* **2010**, *46*, 17–28.
10. Feldman, A.; Peng, J. Automatic detection of idiomatic clauses. In Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing, Samos, Greece, 24–30 March 2013; pp. 435–446.
11. Liu, C.; Hwa, R. Heuristically informed unsupervised idiom usage recognition. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October 2018–4 November 2018; pp. 1723–1731.
12. Sporleder, C.; Li, L. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, Athens, Greece, 30 March–3 April 2009; pp. 754–762.
13. Halliday, M.A.K.; Hasan, R. *Cohesion in English*; Longman: London, UK, 1976; p. 374.
14. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
15. Cilibrasi, R.L.; Vitanyi, P.M.B. The Google similarity distance. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 370–383. [CrossRef]
16. Liu, P.; Qian, K.; Qiu, X.; Huang, X. Idiom-aware compositional distributed semantics. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 1204–1213.
17. Tai, K.S.; Socher, R.; Manning, C.D. Improved semantic representations from tree-structured long short-term memory networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; pp. 1556–1566.
18. Salton, G.D.; Ross, R.J.; Kelleher, J.D. Idiom token classification using sentential distributed semantics. In Proceedings of the 54th Annual Meeting on Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016; pp. 194–204.
19. Williams, L.; Bannister, C.; Arribas-Ayllon, M.; Preece, A.; Spasić, I. The role of idioms in sentiment analysis. *Expert Syst. Appl.* **2015**, *42*, 7375–7385. [CrossRef]
20. Spasić, I.; Williams, L.; Buerki, A. Idiom-based features in sentiment analysis: Cutting the Gordian knot. *IEEE Trans. Affect. Comput.* **2020**, *11*, 189–199. [CrossRef]
21. Flor, M.; Klebanov, B.B. Catching idiomatic expressions in EFL essays. In Proceedings of the Workshop on Figurative Language Processing, New Orleans, LA, USA, 6 June 2018; pp. 34–44.
22. Pike, R. The text editor sam. *Softw. Pract. Exp.* **1987**, *17*, 813–845. [CrossRef]
23. Laurikari, V. NFAs with tagged transitions, their conversion to deterministic automata and application to regular expressions. In Proceedings of the Seventh International Symposium on String Processing and Information Retrieval, La Curuna, Spain, 27–29 September 2000; pp. 181–187.
24. Cox, R. Regular Expression Matching can be Simple and Fast (but is Slow in Java, Perl, PHP, Python, Ruby,...). Available online: https://swtch.com/~{}rsc/regexp/regexp1.html (accessed on 1 January 2021).
25. Gormley, C.; Tony, Z. *Elasticsearch: The Definitive Guide*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2015.
26. Białecki, A.; Muir, R.; Ingersoll, G. Apache Lucene 4. In Proceedings of the SIGIR Workshop on Open Source Information Retrieval, Portland, OR, USA, 16 August 2012; pp. 17–24.
27. Burnard, L. Reference Guide for the British National Corpus (XML Edition). 2007. Available online: http://www.natcorp.ox.ac.uk/docs/URG/ (accessed on 1 January 2021).

28. BNC Consortium. The British National Corpus, Version 3 (BNC XML Edition). Available online: http://www.natcorp.ox.ac.uk/ (accessed on 1 January 2021).
29. Tumblr. I Don't Think Uma will Ever Fully Forgive Mal. Available online: https://tmblr.co/ZVqBcbYOic9sWi00 (accessed on 1 January 2021).
30. Bird, S.; Klein, E.; Loper, E. *Natural Language Processing with Python—Analyzing Text with the Natural Language Toolkit*; O'Reilly Media: Newton, MA, USA, 2009; p. 504.
31. Vega-Moreno, R.E. *Creativity and Convention: The Pragmatics of Everyday Figurative Speech*; John Benjamins Publishing Company: Newton, MA, USA, 2007; p. 264.
32. Langlotz, A. *Idiomatic Creativity: A Cognitive-Linguistic Model of Idiom-Representation and Idiom-Variation in English*; John Benjamins: Amsterdam, The Netherlands, 2006; p. 326.
33. Dutton, K. *Exploring the Boundaries of Formulaic Sequences: A Corpus-Based Study of Lexical Substitution and Insertion in Contemporary British English*; VDM Verlag: Saarbrücken, Germany, 2009; p. 272.
34. Riehemann, S.Z. *A Constructional Approach to Idioms and Word Formation*; Stanford University: Stanford, CA, USA, 2001.
35. Porter, M.F. An algorithm for suffix stripping. *Program* **1980**, *14*, 130–137. [CrossRef]
36. Beke, K. Learn English Today. Available online: https://www.learn-english-today.com/ (accessed on 1 January 2021).
37. Robertson, S.; Zaragoza, H. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.* **2009**, *3*, 333–389. [CrossRef]
38. Clarke, S.J.; Willett, P. Estimating the recall performance of Web search engines. *Aslib Proc.* **1997**, *49*, 184–189. [CrossRef]
39. Cohen, J. A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.* **1960**, *20*, 37–46. [CrossRef]
40. Fleiss, J.L.; Cohen, J.; Everitt, B.S. Large sample standard errors of kappa and weighted kappa. *Psychol. Bull.* **1969**, *72*, 323–327. [CrossRef]
41. Altman, D.G. *Practical Statistics for Medical Research*; Chapman and Hall/CRC: Boca Raton, FL, USA, 1990; p. 624.
42. Richards, J.C.; Rogers, T.S. *Approaches and Methods in Language Teaching*; Cambridge University Press: Cambridge, UK, 2006.
43. Li, Y.; Yosinski, J.; Clune, J.; Lipson, H.; Hopcroft, J. Convergent learning: Do different neural networks learn the same representations? In Proceedings of the 1st NIPS International Workshop on Feature Extraction: Modern Questions and Challenges, Montréal, QC, Canada, 11–12 December 2015; pp. 196–212.
44. Price, L.; Thelwall, M. The clustering power of low frequency words in academic Webs. *J. Assoc. Inf. Sci. Technol.* **2005**, *56*, 883–888. [CrossRef]
45. Schönhofen, P.; Benczúr, A.A. Exploiting extremely rare features in text categorization. In Proceedings of the 17th European Conference on Machine Learning, Berlin, Germany, 18–22 September 2006; pp. 759–766.
46. Fadaee, M.; Bisazza, A.; Monz, C. Examining the tip of the iceberg: A data set for idiom translation. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018; pp. 925–929.