International Journal of Computational Intelligence Systems
Vol. **13(1)**, 2020, *pp.* 904–913
DOI: https://doi.org/10.2991/ijcis.d.200602.002; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Research Article

A Human-Machine Language Dictionary

Fei Liu*, Shirin Akther Khanam, Yi-Ping Phoebe Chen

Department of Computer Science and Information Technology, La Trobe University, Melbourne, Australia

ARTICLE INFO

Article History

Received 18 Dec 2019
Accepted 07 May 2020

Keywords

Text mining
Natural language processing
Knowledge representation

ABSTRACT

In this paper, we propose a framework for building a human-machine language dictionary. Given a concept/word, an application can extract the definition of the concept from the dictionary, and consequently "understand" its meaning. In the dictionary, a concept is defined through its relations with other concepts. Relations are specified in the machine language. To a certain degree, the proposed dictionary has a resemblance to WordNet, which consists of a set of concepts/words with synonyms being linked to form the net. WordNet plays an important role in text mining, such as sentiment analysis, document classification, text summarization and question answering systems, etc. However, merely providing synonyms is not sufficient. The proposed dictionary provides a definition for each concept. Based on the definition, the application can accurately estimate the distance and similarity between concepts. As a monotonic mapping, the algorithm for estimating distances and similarities is proved to be always convergent. We envisage that the dictionary will become an important tool in all Text Mining disciplines.

1. INTRODUCTION

The proposed human-machine language dictionary is based on the Natural Language Independent Knowledge Representation (NLIKR) scheme which will be introduced in the paper. NLIKR defines a concept without using a human language, and the concept is represented/defined through its relations with other concepts. The set of relations that the concept possesses, becomes the definition of the concept. For instance, "the color of water is transparent" describes the relation between 'water' and 'transparent' bound by 'color.' This relation reveals/defines a character of 'water.' Similarly, there are relations such as 'water is a liquid' and 'the boiling point of water is 100°C,' both of which describe the characteristics of 'water.' There may be millions of relations concerning 'water.' These relations collectively constitute the definition of 'water.'

Definitions for all concepts form the online human-machine language dictionary. The dictionary consists primarily of two components: the concept space (**CS**) and the extraction engine (**EE**). Whilst the **CS** contains concepts and their definitions in a hierarchical structure, **EE** accepts queries, extracts data from the **CS** to answer the query. For instance, to understand the meaning of 'water,' a computer application can send 'water' as a query to the dictionary. Receiving the query, the **EE** extracts all relations that 'water' possesses and returns the extracted data as the answer. As a result, the computer application understands that 'water' is a liquid of transparent color, no taste and 100°C boiling point, etc. In other words, the definition of 'water' can be loaded into the application. For the application, 'water' is no longer just a string, but a string that represents certain information. Figure 1 demonstrates this.

WordNet [1], a lexical database of English words, is probably one of the most significant attempts by linguisticians and computer scientists to define English words using synonyms. Words are grouped into synsets. Each synset expresses a distinct concept. Synsets are linked by lexical relations. This effectively reduces the lengthy English definitions that cannot be understood by machines. However, synsets are not sufficient to define a word (concept). Furthermore, a word, which represents an existence, may exhibit its properties in multiple aspects (such as its molecular structure, physical and chemical properties, its source and usages, and its impact on other existences, etc.). None of these can be expressed in WordNet.

Aimed to represent human common knowledge, ConceptNet [2] was created as a semantic graph with words and common phrases linked by relations. ConceptNet is capable of providing a large set of common knowledge that a computer application needs when analyzing natural language-based text and conducting reasoning. The knowledge base consists of over 1.6 million assertions. ConceptNet is automatically generated from the 700,000 sentences of the Open Mind Common Sense Project—a World Wide Web-based collaboration with over 14,000 authors. However, ConceptNet has a serious limitation: its relations are rigid and narrow. One cannot take a word/phrase of their choice and make it a relation. In ConceptNet 5.0, there are limited number of relations (such as isA, relatedTo, partOf, createdBy and definedAs, etc.) being included. Separating relations from words/phrases restricts the expressiveness of the framework, as in the real world, relations can be far more

* *Corresponding author. Email: f.liu@latrobe.edu.au*

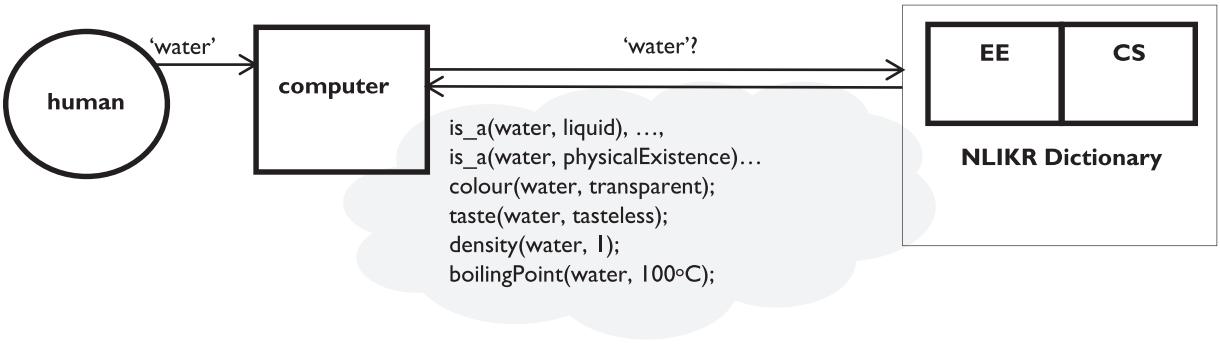

Figure 1 | Extracting the definition of "water" from the Natural Language Independent Knowledge Representation (NLIKR) dictionary.

diversified than the listed types. This also results the difficulties in estimating the similarity and distance between words/phrases.

In the "Read the Web" project by Carnegie Mellon University, Carlson *et al.* proposed a framework to build a constant language learner named Never-Ending Language Learner (NELL) [3]. NELL extracts information from the web to build up a structured knowledge base. NELL also has the capability to learn from its previous learning experience and constantly enhance its performance. After its 10 years' learning experience, NELL has established an extensive knowledge base of 90 million candidate beliefs. However, it also faces difficulties in various aspects. Firstly, since NELL learns from the World Wide Web, any gained knowledge needs to be verified. The verification process requires human participation and may not always be reliable. Furthermore, it is difficult for NELL to distinguish between essential and non-essential knowledge. For instance, it is important to learn "'royal gala' is a type of 'apple'," but it is undesirable to include information such as "the price of 'royal gala' is \$4.95/kg in the supermarket." Resolving inconsistent information is another challenge in the learning process. While the price of 'royal gala' is \$4.95 in the supermarket, the apples are sold at \$3.95/kg in Sam's Grocery Store and \$3.50/kg in the Vegetable and Fruit Wholesale Market. The three different prices represent three pieces of information which are inconsistent. While voluminous, the knowledge base built by NELL is not considered to be powerful due to its lack of inference capability. When scanning through the text, the language learner doesn't have the capability to understand the text. As a result, it doesn't have the capability to infer the fact "royal gala" as a goods can be sold in various venues and in different prices unless the fact is explicitly specified.

The proposed human-machine language dictionary is designed to overcome the problems. Given the concept 'royal gala,' the dictionary is capable of providing information such as 'royal gala' is a sub-concept of 'apple' (i.e. 'royal gala' is a type of 'apple') and it is also a sub-concept of 'goods.' A 'goods' can be sold and has a price. Being a sub-concept of 'goods,' 'royal gala' inherits the properties (i.e. it can also be sold and has a price). Unlike conceptNet, the human-machine language dictionary allows a word/phrase to represent a relation. This significantly increases the flexibility of the knowledge representation and consequently enriches the knowledge base.

From a number of perspectives, there is a resemblance between the **CS** and a lightweight universal ontology [4] which contains concepts and relations, but the two are significantly different. The differences lie in the fact that whilst an ontology aims to classify and match concepts, the purpose of the **CS** is to define them. Ontologies are normally domain-specific, and therefore information-specific. Even for an upper-level ontology which is normally built as a frame to facilitate ontology merging [5], its information inclusion is nevertheless selective. The concentration of ontology research is mainly on concept classification and ontology merging [5,6] rather than concept definition. The **CS**, on the other hand, includes knowledge of all aspects. Its main purpose is to provide definitions for concepts. Being a lightweight and hierarchical structure, its creation can be semi-automatic with little human participation [4]. Methodologies for ontology generation can certainly be adopted for the dictionary creation.

Two types of relations, namely inheritance and association, are included in the **CS**. Inheritance reflects the "⋯is a⋯" relation in which a concept inherits properties from its super concepts. For instance, "water is a type of liquid" reveals an inheritance relation between 'water' and 'liquid' where 'liquid' is a super concept of 'water.' Association refers to connection, interaction and comparison between concepts. For instance, "the color of water is transparent" expresses an association between 'water' and 'transparent.' It is the relations between 'water' and other concepts such as (being a type of) 'liquid,''transparent' (in color) and 'tasteless' (in taste), etc., define characteristics of 'water.' As a result, the definition of 'water' can be expressed as a set of relations without the involvement of a human language. Figure 2, which is a segment of the **CS**, briefly illustrates this.

In Figure 2, each oval represents a concept. A dashed arrow symbolizes inheritance while a solid arrow represents an association. Concepts are classified and stored in a hierarchical structure where 'existence' is the top of the structure. 'existence' is divided into 'physicalExistence' and 'abstractExistence.''physicalExistence' and 'abstractExistence' are sub-concepts (children) of 'existence.' 'existence' is the super-concept (parent). The hierarchical structure can be multi-dimensional. That is, while 'physicalExistence' can be 'lifeExistence' and 'noneLifeExistence,' it can also be a super-concept of solid,'liquid' and 'gas.' A concept inherits properties of its super concepts. In Figure 2, 'liquid' is a super concept of 'water.' Therefore, 'water' inherits properties/characteristics of 'liquid.'

We can search on the Google search engine to obtain a large volume of information about 'water,' but the obtained information is text-based. It can be viewed and understood by humans, not by a computer. Consequently, automatic information processing

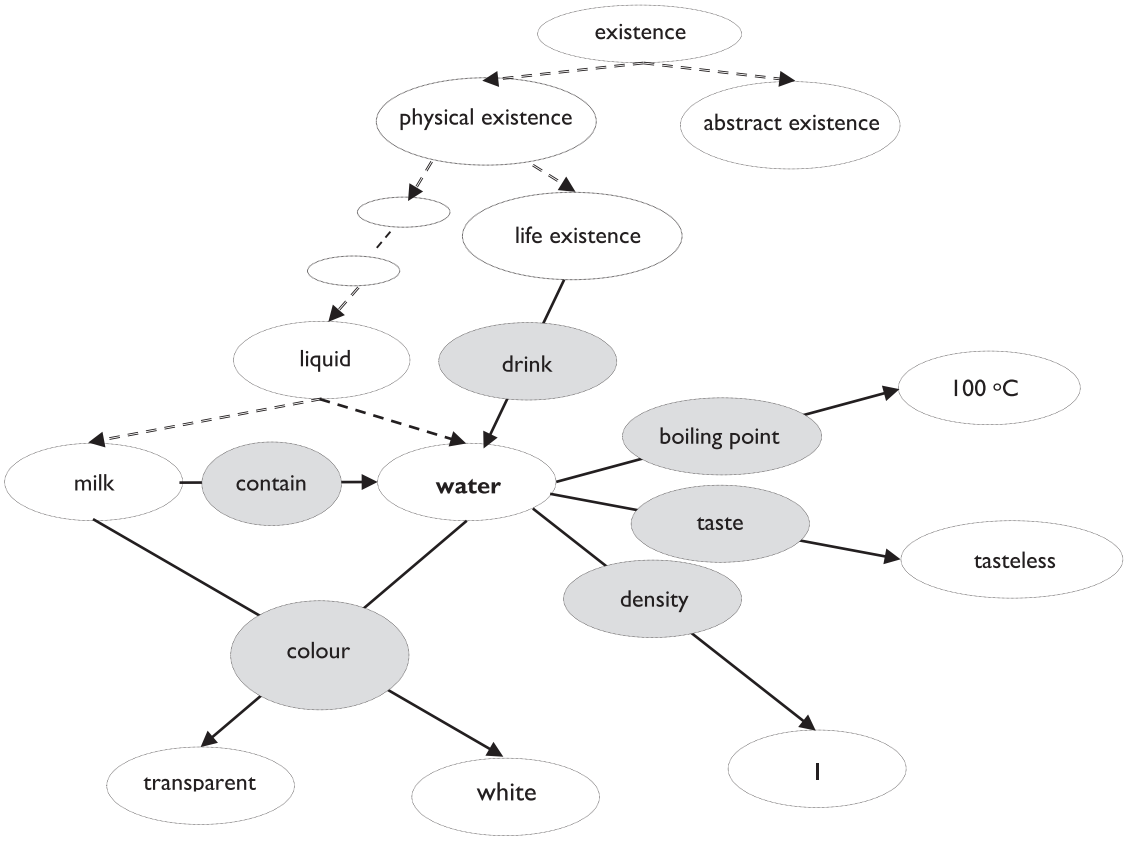

Figure 2 | A segment of C.

cannot be performed. Substantial research in text mining [2,7,8], such as sentiment analysis, question-answering systems and semantic social networks, has been conducted in the attempt to extract information from human-language-based text. However, methodologies in these areas, whether supervised or unsupervised, cannot achieve 100% accuracy. This is because these methodologies analyze the key words in the text to ESTIMATE its meaning, rather than UNDERSTANDING it.

The center of the research is (1) human languages are not suitable media for knowledge representation in a machine. It is necessary that human knowledge is translated into a machine language to enable automated information processing across domains and platforms and (2) translation can be achieved through a scheme. We propose NLIKR as such a scheme.

The contribution of the research is that the NLIKR dictionary provides the definition of a concept by extracting the relations between the concept and other concepts. Most importantly, the definition is expressed as a set of tuples, not a piece of human language-based text. This enables an application to accurately "understand" the definition.

The rest of the paper is organized as follows: Section 2 briefly discusses previous research. Section 3 presents the definition of NLIKR and its theoretical foundation. Section 4 discusses two important algorithms to estimate the distance and similarity. The construction of the dictionary and some experiment on the framework are briefly discussed in Section 5. Finally, Section 6 concludes the paper and suggests directions for further research.

2. PREVIOUS RESEARCH

NLIKR is inspired by the pixel representation of images. When the pixel representation was invented in the 1960s, people were amazed by the fact that thousands of pixels are required to represent a single image. After a half-century, pixel representation has been proven to be the most effective mechanism in image representation. The success of the technology can be attributed to the fact that the representation is able to divide an image into the most fundamental element, the pixel. When it comes to knowledge representation, the question becomes what is the most fundamental element in describing things (concepts)? We believe it should be the relation, i.e., relations between concepts define the characteristics of a concept.

Research on knowledge representation has a long history in which a number of significant mechanisms have been developed. These methods are classified as logic-based [9], semantic networks, frames, scripts and ontology-based mechanisms [10].

In logic-based knowledge representation, information is represented in the clause (Horn clause) form to facilitate reasoning. The scheme is effective in supporting logic programming (i.e., reasoning based on first-order logic). It is easy to use, however, has limited expressiveness.

Developed in the 1950s, semantic networks represent knowledge by defining concepts and their semantic relations. The scheme can be illustrated by using a graph where concepts are defined as nodes and nodes are linked by edges that represent relations. Much of the research in this area is implemented in LISP. The scheme is pow-

erful in terms of its expressiveness, however its definitions of concepts and relations heavily depend on natural languages, and this seriously disadvantages its implementation.

The 1970s witnessed the development of frame-based knowledge representation. Many believe that the frame-based scheme also inspired the establishment of object-oriented software development. In the frame-based scheme, related data are grouped and represented as a frame. For instance, street number, street name, city, state, country and postcode can fit into a frame and is named postal address. The scheme was implemented as MIKE in PROLOG, and it has been regarded as an effective mechanism in knowledge representation.

The Semantic Web inspired a new wave of research in the area. Research on the Semantic Web explores possible approaches for web information representation and extraction. RDF [11] and OWL [12] are the two major frameworks in this area. The frameworks which are XML-based have been effective for concept classification and matching rather than concept definition.

Aiming to represent common sense knowledge, ConceptNet [2] was introduced in 2004. The establishment of ConceptNet was largely based on CYC knowledge base [13,14]. Compared with CYC, ConceptNet is well advanced. This is because (1) ConceptNet is structural, that is, all its concepts are stored in a hierarchical structure instead of being a set of inference rules as CYC expressed. Relations between concepts are also included; (2) unlike WordNet or CYC which are handcrafted, ConceptNet can be built automatically. However, the purpose of ConceptNet is NOT to represent words/-concepts, but to express common sense knowledge. As a result, definitions of concepts in the structure are not only incomplete, but also irregular. In some circumstances, concepts are not words, but phrases which represent scenarios, situations or events. The concentration of the framework is not on defining English words, but to reveal the relations between concepts, so that a piece of commonsense knowledge can be conveyed.

3. DEFINITION OF NLIKR

NLIKR is presented in this section. Estimation of the distance and similarity under the scheme are also discussed.

3.1. Concept Space CS—The Hierarchical Structure

In NLIKR, each existence is a concept. For instance, 'cat,' 'philosophy,' 'move,' 'black' and 'peacefully' are all concepts. The set of all concepts is the **CS** which is a huge hierarchical structure formed by concepts being bound in two types of relations: inheritance and association.

Definition 1. In NLIKR, each existence (physical or abstract) is a concept (denoted as c). All concepts form a set. The set is named **CS**. **CS** = $\{c_1, c_2, \ldots, c_n\}$ (n is a finite integer and n > 0).

Each concept is an element of the **CS**. The **CS** has a finite number of elements.

Definition 2. Let c_1 and c_2 be two concepts in **CS**. c_1 is defined as a sub-concept (descendant) of c_2 (denoted as $c_1 \subseteq c_2$) if c_1 is a type of c_2, in which case, c_2 is called a super-concept (ancestor) of c_1.

For instance, 'lemon' is a sub-concept (descendant) of 'citrus' and 'fruit.' 'fruit' is the super-concept (ancestor) of 'citrus' and 'lemon.' A concept inherits properties of its super concepts. The CS is unique in terms of its hierarchical structure.

The inclusion operator $\subseteq$ is transitive. That is, for concepts c1, c2 and c $\in$ **CS**, c1 $\subseteq$ c2 and c2 $\subseteq$ c infers c1 $\subseteq$ c. Being a sub-concept of c2, c1 possesses (inherits) properties/characteristics of c2. This means that $\forall$d and e $\in$ **CS** if c1 $\subseteq$ c2 and there is an association <c2, d, e> then the association <c1, d, e> also exists. An example is that 'vehicle' is a super-concept of 'car' and therefore 'car' exhibits characteristics of 'vehicle.' As '<vehicle,' 'has,' 'wheel>' exists, '<car,' 'has,' 'wheel>' also holds.

It is obvious that $\subseteq$ is an order on **CS**, hence <**CS**; $\subseteq$> is an ordered set [15]. We will now closely examine the structure of the **CS** and the two types of relations: inheritance and association.

3.1.1. Inheritance—creation of the hierarchical structure

Inheritance reflects the "$\cdots$***is a***$\cdots$" relation. It refers to the phenomenon that an association possessed by the super-concept is also possessed by the sub-concept.

Inheritance can be multi-dimensional. This refers to the fact that a concept can be divided in different ways. For instance, the 'human' concept can have sub-concepts such as 'infant,' 'child,' 'adolescent,' 'adult' and 'elderly' which divide the 'human' concept based on age. It can also have the sub-concepts 'woman' and 'man' which split the concept into two distinctive categories. Furthermore, 'human' individuals can also be identified by their nationality. As a result, 'human' may have 195 offspring such as 'australian,' 'brazilian,' 'canadian,' $\cdots$etc.

3.1.2. Association—establishment of properties

An association between concepts reflects a property/characteristic of the concepts. For instance, 'student' is associated with 'educationalOrganization' through 'enrolment.' The triple '<student,' 'enrolment,' 'educationalOrganization>' forms an association and this association reflects a characteristic of 'student.' Of course, it also reveals a characteristic of 'educationalOrganization' and 'enrolment.'

Each concept may have associations with millions of concepts. These associations describe the concept and establish the properties of the concept. For example, '<water,' 'color,' 'transparent>' and '<water,' 'taste,' 'tasteless>' define the physical properties of 'water.' As a result, an application can refer to these associations when processing a text that contains the string 'water.' Consequently, the application's understanding of 'water' is far beyond the simple string W-A-T-E-R. Not only is the application aware of the physical and chemical properties of 'water,' it also possesses information about the sources, usages and applications of the concept. Ultimately, we let concepts describe/define each other in a machine language.

3.1.3. End concepts

At this stage, a question arises: if we expect concepts to define each other, in which way can precise values be obtained? For instance, the association '<milk,' 'color,' 'white>' defines the color of milk, which is 'white.' But precisely, what is 'white'? In which way should 'white' be represented? 'white' as a color can be represented precisely as a triple (256, 256, 256) in the **CS**. Each number represents the level of 'red,' 'green' or 'blue' components. 'white' is named an end concept.

Definition 3. A concept that can be defined by a set of values and that does not have any sub concept is an end concept.

Also referred to as a terminal concept in ontology, an end concept can be precisely defined by a value or a sequence of values. End concepts play a vital role in concept definition. This is because by simply letting concepts define concepts without any quantified information, the definitions may become a set of definition loops. Such definitions may not be valuable.

3.1.4. Abstract concepts

A concept can be abstract. An abstract concept does not have a model in the real world. Instead, it represents a collection of concepts. In **CS**, an abstract concept normally acts as a placeholder to represent a type of concepts. For example, 'educationalOrganization' is an abstract concept An 'educationalOrganization' can be a 'kindergarten,' 'primarySchool,' 'middleSchool,' 'vocationalTrainingCollege' or 'university,' but an 'educationalOrganization' itself is not an object in the real world. Meanwhile, the existence of 'educationalOrganization' in **CS** is important. It acts as the parent of all types of schools and universities, and it has its own characteristics.

The question now becomes, how many concepts and how many associations should be included in **CS**? The answer is all concepts and their associations should be included. Only in this way, human knowledge about things can be completely translated into machine knowledge.

3.2. Distance, Similarity and Synonyms

The distance of concepts is a measure of the closeness of the two, while the similarity of concepts represents the resemblance of the characteristics. These are two distinctive measurements and should not be used as the inverse of each other. Two concepts can be closely related, but not similar. For instance, 'snapper' and 'water' are closely related in the sense that a 'snapper' lives in 'water,' breathes through 'water' and eats from 'water.' However, 'snapper' and 'water' are not similar at all.

Measuring the similarity of concepts has a wide range of applications in various areas, such as word sense disambiguation, question-answering systems and sentiment analysis etc. If two concepts are sufficiently similar, they are synonyms. A substantial amount of work has been conducted on measuring the distance and similarity of words. In WordNet [1], for instance, the similarity of words is estimated based on the number of edges between the words [16]. In sentiment analysis, to determine the sentiment orientation of an adjective, one could measure the distances of the word to the two polarities, good and bad, and determine the orientation based on the shorter distance [7,8]. The distance between words can also be estimated through a Google search [17]. Estimating the similarity of concepts has also been a discipline of ontology research [18]. While the similarity of concepts can be calculated based on the hierarchical positions, it can also be estimated based on features across multiple ontologies [5,6]. The methodologies, however, can lack accuracy in some circumstances. Furthermore, it is difficult to estimate the strength of the sentiment through WordNet. In most situations, similarity and distance are intermingled.

In this section, we will attempt to define the distance and similarity between two concepts based on their positions in **CS**, the associations between them and their common associations with other concepts. This allows both measurements to be calculated based on their semantic contents. Therefore, the calculation is more likely to be accurate.

3.2.1. Distance and similarity

As previously indicated, a concept in **CS** is defined from two perspectives, vertically by inheritance and horizontally by its associations with other concepts. Consequently, a number of factors, such as the common ancestors and common associations of the two concepts, as well as the association bindings between the concepts will be utilized in measuring the distance and similarity.

1. Percentage of Common Ancestors (PCA)

The percentage of common ancestors is the ratio of common ancestors and all ancestors of the two concepts. It is an important measure of the similarity. The higher the percentage, the higher the similarity.

Definition 4. Let $c \in \mathbf{CS}$ be a concept. The set of ancestors of c, $A(c) = \{e \mid e \in \mathbf{CS} \text{ and } c \subseteq e\}$ is a set of super-concepts of c.

Given two concepts $c_1 \in \mathbf{CS}$ and $c_2 \in \mathbf{CS}$, it is not difficult to see that the intersection of the two sets, $A(c_1) \cap A(c_2)$, contains ancestors of both c_1 and c_2.

Definition 5. Let c_1 and c_2 be two concepts in **CS**. The percentage of common ancestors of c_1 and c_2 is defined as

$$PCA(c_1, c_2) = \frac{2|A(c_1) \cap A(c_2)|}{|A(c_1)| + |A(c_2)|} \quad (1)$$

where $|A(c_1)|$, $|A(c_2)|$ and $|A(c_1) \cap A(c_2)|$ are cardinalities of $A(c_1)$, $A(c_2)$ and $A(c_1) \cap A(c_2)$. Obviously, for any c_1 and c_2, $0 \leq PCA(c_1, c_2) \leq 1$. A high percentage of common ancestors is an indication of a short distance and high similarity.

2. Percentage of Common Associations (PCAS)

The percentage of common associations represents the ratio of the same associations and all associations the two concepts possess. It indicates the resemblance of the concepts.

Definition 6. Let $c \in \mathbf{CS}$, the set AS(c) is the set of associations of c. Given $c_1 \in \mathbf{CS}$ and $c_2 \in \mathbf{CS}$, if both $<c_1, d, e>$ and $<c_2, d, e>$ are associations of **CS** ($d \in \mathbf{CS}$ and $e \in \mathbf{CS}$), then the association is a common association of c_1 and c_2, denoted as $<(c_1, c_2), d, e>$. The set of all common associations of c_1 and c_2 is denoted as $CAS(c_1, c_2)$.

Definition 7. Let c_1 and c_2 be two concepts of **CS**. The percentage of the common associations for the two concepts is

$$PCAS(c_1, c_2) = \frac{2\,|CAS(c_1, c_2)|}{|AS(c_1)| + |AS(c_2)|} \tag{2}$$

For two concepts, a common association is an association with the same concept and has the same property. For instance, A_1: '<water,' 'color,' 'transparent>' and A_2: '<oxygen,' 'color,' 'transparent>' are a common association of 'water' and 'oxygen.' Again, for any c_1 and c_2, $0 \leq PCAS(c_1, c_2) \leq 1$. A high percentage of common association is also an indication of a short distance and high similarity.

3. Percentage of Binding Association (PBAS)

A binding association is an association linking two concepts. Obviously, a high percentage of binding associations is an indication of the closeness of the concepts.

Definition 8. Let c_1, c_2 and p be concepts of CS. The association <c_1, p, c_2>, if it exists, is named a binding association between c_1 and c_2. The set BA(c_1, c_2) contains all binding associations between c_1 and c_2. The percentage of the binding association between c_1 and c_2 is

$$PBAS(c_1, c_2) = \frac{2\,|BA(c_1, c_2)|}{|AS(c_1)| + |AS(c_2)|} \tag{3}$$

Clearly, $0 \leq PBAS(c_1, c_2) \leq 1$.

Given two concepts, a binding association represents a connection or interaction between them. For instance, A3: 'fantail' and 'water' have an association via 'liveIn' (i.e., Fantail, as a type of fish, lives in water). A3 is a binding association between 'fantail' and 'water.' Between the two concepts, there are other binding associations via 'breathFrom' (A4) and via 'drink' (A5). A high percentage of binding associations suggests the closeness of the two concepts, and hence a short distance. Meanwhile, a high percentage of binding association is not necessarily an indication of a high similarity.

It is not difficult to see that in terms of common ancestors, 'fantail' is closer to 'spoodle' than it is to 'water.' This is because the common ancestor of 'fantail' and 'water' is 'existence' while for 'fantail' and 'spoodle,' the common ancestors are 'animal,' ⋯'lifeExistence,' 'existence.' However, when it comes to common and binding associations, 'fantail' is closer to 'water' due to multiple bindings such as A3, A4 and A5.

As binary relations, PCA/2, PCAS/2 and PBAS/2 are symmetric regardless of the positions of c_1 and c_2 in the CS. That is for any concepts c_1 and c_2

$$PCA(c_1, c_2) = PCA(c_2, c_1) \tag{4}$$

$$PCAS(c_1, c_2) = PCAS(c_2, c_1) \tag{5}$$

$$PBAS(c_1, c_2) = PBAS(c_2, c_1) \tag{6}$$

We understand that any two concepts have at least one common ancestor, as all concepts are descendants of 'existence.' Hence, inheritance can always be utilized as a measure in calculating the distance and similarity.

Another example is the distance between 'human' and 'thought' which is supposed to be greater than that of 'human' and 'paramecium.' This is because the common ancestor of 'human' and 'thought' is 'existence' while the common ancestors of 'human' and 'paramecium' are 'animal,' 'physicalExistence' and 'existence.' However, this is not necessarily the case, as 'human' is associated with 'thought' through 'possess' (i.e., a human may possess thoughts) and the association binding reduces the distance between 'human' and 'thought.' Below are the definitions of the distance and similarity between concepts.

Definition 9. Let c_1 and c_2 be two concepts. The distance of c_1 and c_2

$$D(c_1, c_2) = w_{CA} \log \frac{1}{PCA(c_1, c_2)} + w_{CAS} log \frac{1}{PCAS(c_1, c_2)} + w_{AB} PBAS(c_1, c_2) \tag{7}$$

where w_{CA}, w_{CAS} and w_{BAS} are three weights. $w_{CA} \geq 0$, $w_{CAS} \geq 0$ and $w_{BAS} \geq 0$.

As a distance, D/2 is non-negative (i.e., $D(c_1, c_2) \geq 0$); reflexive (i.e., $D(c_1, c_1) = 0$); symmetric (i.e., $D(c_1, c_2) = D(c_2, c_1)$) and subadditive (i.e., $D(c_1, c_3) + D(c_3, c_2) \geq D(c_1, c_2)$).

The calculation of the similarity is based on common ancestors and common associations. The association binding is irrelevant.

Definition 10. Let c_1 and c_2 be two concepts. The similarity of c_1 and c_2

$$S(c_1, c_2) = f_{CA} PCA(c_1, c_2) + f_{CAS} PCAS(c_1 c_2) \tag{8}$$

where f_{CA} and f_{CAS} are two coefficients satisfying $f_{CA} \geq 0$ and $f_{CAS} \geq 0$ and $f_{CA} + f_{CAS} = 1$

Although the distance can be an arbitrary positive number, the similarity of two concepts is a real number within the [0, 1] interval.

3.2.2. Synonyms

Synonyms are concepts that have a very high similarity. Meanwhile, a short distance is not necessarily an indication of synonyms. As the previous example indicates, the distance between 'water' and 'snapper' is short as a 'snapper' lives in 'water,' breathes through 'water' and eats from 'water'; but 'water' and 'snapper' are not similar and should not be regarded as synonyms. Identifying synonyms is mainly based on the hierarchical structure, therefore PCA/2 is the main factor. In other words, two synonyms must be in the same position in the **CS** and have identical sets of ancestors.

Definition 11. Let c_1 and c_2 be two concepts. c_1 and c_2 are defined as synonyms if and only if

$$PCA(c_1, c_2) * PCAS(c_1, c_2) > \alpha \tag{9}$$

where $\alpha < 1$ and $\alpha \approx 1$ is a positive value.

4. ALGORITHMS

This section presents two algorithms: an algorithm to identify all ancestors and an algorithm to search for all associations for a concept. These algorithms will facilitate the calculation of PCA, PCAS *and* PBAS between two concepts.

4.1. CS—A Complete Lattice

As indicated previously, all concepts form a **CS**. **CS** is a hierarchical structure bound by inheritance. **CS** is finite. It was also pointed out that $\subseteq$ is an order on **CS**, and therefore < **CS**; $\subseteq$> is an ordered set. In this section, we will further examine the CS.

Definition 12. In the ordered set <**CS**; $\subseteq$>, the concept 'existence' (denoted as **E**) is defined as the super-concept of all concepts. $\forall$ c $\in$ <**CS**; $\subseteq$>, c $\subseteq$ **E**. **E** is the upper bound of <**CS**; $\subseteq$>. Similarly, the concept void (denoted as **V**) is defined as the sub-concept of all concepts. $\forall$ c $\in$ < **CS**; $\subseteq$>, **V** $\subseteq$ c. **V** is the lower bound of <**CS**; $\subseteq$>.

Since E $\in$ <**CS**; $\subseteq$>, **E** is the least upper bound of <**CS**; $\subseteq$>. Similarly, **V** is the greatest lower bound of the ordered set. The ordered set <**CS**; $\subseteq$> is like a downward tree structure. Each concept is a node of the tree, with the least upper bound being the root and the greatest lower bound being the leaf of the tree.

Definition 13. Let c_1 and c_2 be two elements of the ordered set <**CS**; $\subseteq$>, c_2 is the parent of c_1 if (1) $c_1 \subseteq c_2$ and (2) $\nexists c$ such that $c_1 \subseteq c$ and $c \subseteq c_2$. In which case, c_1 is a child of c_2.

A parent concept is a direct super concept. A child concept is a direct sub concept.

Definition 14. Let c_1 and c_2 be two elements of <**CS**; $\subseteq$>, the join of c_1 and c_2,$c_1 \vee c_2$ is an element of <**CS**; $\subseteq$> that satisfies the following conditions:

$c_1 \subseteq c_1 \vee c_2$ and

$c_2 \subseteq c_1 \vee c_2$ and

$\nexists c \in$ < **CS**; $\subseteq$> such that $c_1 \subseteq c$ and $c_2 \subseteq c$ and $c \subseteq c_1 \vee c_2$

The meet of c_1 and c_2,$c_1 \wedge c_2$ is an element of <**CS**; $\subseteq$> satisfying the following conditions:

$c_1 \wedge c_2 \subseteq c_1$ and

$c_1 \wedge c_2 \subseteq c_2$ and

$\nexists c \in$ <**CS**; $\subseteq$> such that $c \subseteq c_1$ and $c \subseteq c_2$ and $c_1 \wedge c_2 \subseteq c$

The join of c_1 and c_2 is the smallest common super concept of c_1 and c_2. The meet of c_1 and c_2 is the largest common sub concept of c_1 and c_2.

The join of **CS**, $\vee$**CS** is the join of all elements of **CS**. The meet of **CS**, $\wedge$**CS** is the meet of all elements of **CS**.

Theorem 1: *The ordered set <**CS**; $\subseteq$> is a complete lattice. A monotonic mapping on the lattice converges to a fixpoint.*

*The proof of the theorem is straightforward. Let $c_1 \in$ <**CS**; $\subseteq$> and $c_2 \in$ <**CS**; $\subseteq$> be two elements of the ordered set. Since $c_1 \vee c_2 \in$ <**CS**; $\subseteq$> and $c_1 \wedge c_2 \in$ <**CS**; $\subseteq$>, <**CS**; $\subseteq$> is a lattice. Furthermore, since $\vee$**CS** = **E** and $\wedge$**CS** = **V** and **E** $\in$ <**CS**; $\subseteq$> and **V** $\in$ <**CS**; $\subseteq$>, <**CS**; $\subseteq$> is a complete lattice.*

*According to the Fixed-point Theory [15], a monotonic mapping on <**CS**; $\subseteq$> converges to a fixed-point.*

The theorem ensures the convergence of the algorithms presented below.

4.2. The Search for Ancestors Algorithm

The algorithm to search for ancestors can be recursion or iteration based. For the recursion-based algorithm, the search starts from the current concept. The algorithm returns parent concepts of the current concept, and then applies the search algorithm to parent concepts. The recursion is terminated at the stage where no further parent concepts can be identified. Meanwhile, the iteration-based algorithm also starts from the current concept. It places the current concept in an array, places the parents of the current concept in the array, the parents of the parents of the current concept in the array, etc. Eventually, it returns the array. Below is the algorithm to search for the ancestors of a concept.

Algorithm 1: SEARCH_ANCESTORS (thisConcept)

```
INPUT thisConcept
OUTPUT ListOfAncestors
1: LET ListOfAncestors = [thisConcept]
2: LET currentConcept = 0
3: REPEAT
4: numberOfAncestors = ListOfAncestors.length()
5: FOR(index = currentConcept TO ListOfAncestors.length()-1 WITH
   index++)
6: BEGIN
7: CONCATINATE(ListOfAncestors,
   FIND_ALL_PARENTS(ListOfAncestors[currentConcept]))
8: currentConcept++
9: END
10: UNTIL numberOfAncestors == ListOfAncestors.length()
11: return ListOfAncestors
```

The algorithm is iteration-based. FIND_ALL_PARENTS/1 returns a list of concepts that are immediate ancestors (parents) of the parameter. CONCATINATE/2 takes two lists as parameters. It concatenates the second parameter into the first.

Note that the recursive version of the algorithm can also be defined. The list of ancestors can be obtained by repeatedly identifying the list of parents for a concept.

4.3. The Identify Common Association Algorithm

The percentage of common associations of two concepts is an important indication of the similarity of the two. The higher the percentage, the more similar they are. For synonyms, their percentage should be almost 100%. This reflects the fact that synonyms have basically the same associations.

The algorithm to identify common associations is presented below. In the algorithm, ASSOCIATIONS/1, IS_ANCESTOR/2 and ADD/3 are three functions/methods. ASSOCIATIONS/1 extracts and returns associations of the parameter. Each association is represented by its subject, predicate and object. IS_ANCESTOR/2

Algorithm 2: IDENTIFY_COMMON_ASSOCIATION (concept1, concept2)

```
INPUT concept1, concept2
OUTPUT ListOfCommonAssociations
1: LET Associations1 = ASSOCIATIONS(concept1)
2: LET Associations2 = ASSOCIATIONS(concept2)
3: LET ListOfCommonAssociations = { }
4: FOR(index1 = 0 TO Associations1.length()-1 WITH index1++)
5: FOR(index2 = 0 TO Associations2.length()-1 WITH index2++)
6: IF((Associations1[index1].predicate == Associations2[index2].predicate
   OR
   IS_ANCESTOR(Associations1[index1].predicate,
   Associations2[index2].predicate)
   OR
   IS_ANCESTOR(Associations2[index2].predicate,
   Associations1[index1].predicate)) AND
   (Associations1[index1].object == Associations2[index2].object OR
   IS_ANCESTOR(Associations1[index1].object,
   Associations2[index2].object) OR
   IS_ANCESTOR(Associations2[index2].object,
   Associations1[index1].object))
7: THEN
8: ADD(Association1[index1], Association2[index2],
   ListOfCommonAssociations)
9: RETURN ListOfCommonAssociations
```

returns TRUE if its first parameter is an ancestor of the second parameter, FALSE otherwise. ADD/3 takes three parameters. Parameter one and two are associations while parameter three is a list of associations. ADD/3 simply adds the first two parameters to the third.

NLIKR has not yet been implemented. The difficulties of the implementation are not on building its EE, but more likely on the construction of the **CS**, which represents a huge amount of work if constructed manually. Consequently, we aim to automate/semi-automate the process, that is, to build **CS** based on automated/semi-automated text learning.

Although the purposes and structures are different, the construction of **CS** can be based on algorithms and tools for building ontologies. It has been suggested that techniques for ontology learning can be classified into three categories: statistics-based, linguistics-based and logic-based [4]. The second category is particularly important for the dictionary construction. This is because **CS** presents a set of facts. It is simply a description of entities and relationships between entities in the machine language. Clustering or concurrence analysis is not required at this stage. Neither is it necessary to apply logic inference.

5. EXPERIMENTS

Implementation of a prototype of the dictionary based on existing thesaurus repositories and ontologies has been performed by Khanam *et al.* [19]. Data are imported from various sources such as DBpedia, WordNet and ConceptNet in the CSV format, compared and merged to form the structure. The implementation process is conducted in a semi-automatic manner.

5.1. Defining Concept Using Short Abstract

Each word is defined using short abstract as characteristics or relation. This short abstract as a characteristic or relation are imported from Dbpedia, CRISP or conceptNet. There are a few steps to import short summary as the definition of a concept into the dictionary. These steps are described as below

- Pre-processing content: short abstracts are required to pre-processed before importing into ontology because the sentences in short abstract may be too complex to relate with other word.
- Locating or adding concept for defining concept into ontology: this step confirms that all the words of sentences in abstract has been presented in our ontology.
- Building relation for a sentence of concept: relationships among the word in sentences are made by joining the words and declared this relation as superclass of concept.

While importing short abstract, the dictionary is enriched by adding further concepts, relations or hypernyms using wordnet, Dbpedia and conceptNet.

5.2. Importing Instances of Domain from DBPedia

The procedure of importing instances of concept from of DBpedia data are given. All properties name has been assigned as concept in our ontology and corresponding instances data are added to ontology from DBpedia sources. Finally, instances of a concept are added from DBpedia data.

5.3. Importing Relation of Domain from ConceptNet to Ontology

ConceptNet use some important relations or phrase to represent concept such as CapableOf, Causes, CreatedBy. This important phrase and data related to this phrase are imported from ConceptNet in our ontology. Words in each relation and corresponding data are represented as concept in ontology. Also relate the concept with relation or phrase according to the conceptNet data in the dictionary.

5.4. Retrieving and Representing Data

This is the stage when proposed ontology are ready for query. In this section, a system or application has been proposed where word can be searched from ontology. In this system, word can be queried as concept or instances to find the definition or information about concept or instances. Finally, question about concept can be asked with attribute to know relation between concept and property. This system will represent the query data in more friendly and informative way.

5.5. Comparison of Other Works with the Scheme

In the experiment [19], ontology base repository is compared with some important resource such as DBpedia, ConceptNet, WordNet.

Table 1 show the comparison of these resources with proposed repository derived from the proposed experiment. The proposed ontology is one repository where concepts are interconnected in many more ways compare to wordnet and Dbpedia. Therefore, data is represented in more meaningful way in NLIKR. Words can be represented with attribute in our ontology. For example, size of elephant is big. This is represented in our ontology for elephant→ (*size, big*). In this sense, the proposed repository represents data in a more meaningful and structured way. The definition of word or information about the word will be represented when a word is searched in system. Also, the properties of superclass as definition will be inherit to class and will be represented while a class or concept is queried.

In relation to implementation details, we believe that concepts and their associated relations can be extracted from human language-based document collections. These concepts and relations can be inserted into **CS** to become elements of the space. A relation can be formed with multiple elements. Consequently, the relation does not necessarily have to be represented as a triple. A multi-ary predicate can be formed to represent the relation, its conditions and constraints. As **CS** is lightweight, a preferred language for its construction can be JSON or XML. Algorithms for extracting relations, identifying common ancestors and estimating the distance and similarity can be implemented in **EE** using a web programming language such as JAVA or JAVA Script.

Compared with other frameworks such as WordNet, ConceptNet and DBPedia, the NLIKR dictionary contains substantially higher volume of data [19]. Therefore, accessing a definition in the dictionary takes longer time than it is normally required in any other knowledge bases despite the differences in extraction mechanisms. In WordNet, one extraction normally includes one or multiple words with their associated linkages, while in ConceptNet, DBPedia or NELL, a single query will normally result one or a number of assertions to be extracted and returned. In the NLIKR dictionary, however, extracting one definition involves identifying all inheritance and association relations which may consist of millions of assertions. This notably reduces the speed of the dictionary [19]. In short, comes to search, the complexity of DBPedia, WordNet and ConceptNet are at the O(n) level while NLIKR is $O(n^2)$ (n is the number of the concepts). Meanwhile, since the dictionary is capable of extracting the whole set of relations, its estimation of the distance and similarity for concepts is far more accurate than what is performed by WordNet.

Table 1 Comparison of NLIKR dictionary with DBPedia, WordNet and ConceptNet [19].

	DBPedia	WordNet	ConceptNet	NLIKR
Meaningful and well-structured properties	N	N	N	Y
All concepts are classes or instances of classes	N	N	Y	Y
Has no specific data properties	N	N	N	Y
Limited object property declaration (two or three)	N	N	N	Y
Supporting multiple types of relationships	N	N	Y	Y
Concepts are linked in various ways	N	N	Y	Y
Include instance information and corresponding data	Y	N	N	Y
Allows queries between two words	Y	N	N	Y
Allows inheritance	N	Y	N	Y
Allows class attributes	N	N	N	Y

NLKIR, Natural Language Independent Knowledge Representation.

5.6. Applications of the NLIKR Dictionary

Comes to applications of the NLIKR directory, we can identify its applications in various areas. For instance, when the query 'Is a spoodle a dog?' is entered into the Google search engine, the answer is 'The spoodle is a hybrid of a cocker spaniel (English or American) and a Poodle (toy or miniature). They have become increasingly popular over the last 10 years. The aim of crossbreeding is to minimize the genetic diseases that can be present in purebred dogs.' The answer is not ideal. The most accurate answer should be simply 'yes.' This is due to the search engine being incapable of recognizing the relations 'a cocker spaniel is a dog' and 'a poodle is a dog.' Google selected the answer based on analyzing the keywords occurring in the text, rather than understanding the text.

Meanwhile, the concept 'spoodle' is defined as a sub-concept of 'dog' in the NLIKR dictionary. By accessing the dictionary, a machine can recognize the fact

isA(spoodle, dog)

Therefore, 'yes' can be immediately returned as the answer. As a second example, when the query 'what is the relationship between spoodle and water?' is entered into the Google search engine, the search engine is unable to answer the query, and therefore returns a sequence of web links containing the keywords 'spoodle' or 'water.' The performance of the search engine is disappointing. By accessing the NLIKR dictionary, a machine can extract information

isA(spoodle, dog)

isA(dog, mammal)

isA(mammal, animal)

isA(animal, lifeExistence)

drink(lifeExistence, water)

bodyComponentOf(water, lifeExistence)

Consequently, the returned answer becomes 'spoodle drinks water' and 'water is a component of the body of a spoodle.'"

We envision that the NLIKR dictionary will find a wide range of applications in various fields [13,20,21] including text mining, natural language processing, user interface design, cloud computing and big data.

6. CONCLUSIONS

We have presented a framework for building a human and machine language translation dictionary. The translation is conducted in the way that each concept, denoted using an English word/phrase, is defined by its relations with other concepts and its position in the CS. The dictionary consists of two components: **CS**, which is a hierarchical structure formed by concepts and their relations and **EE**, which extracts information from **CS**. The dictionary can be an

important tool in facilitating human and machine interactions and research in text mining.

The estimation of distances and similarities has been discussed in this research. We argue that the distance and similarity should be calculated separately since they are distinctive measurements. The distance of two concepts depends on PCA, PCAS and PBA, however, the calculation of the similarity only relies on PCA and PCAS. The distance is used to represent how closely two concepts are related, and the similarity indicates the degree of resemblance between two concepts. An extremely high similarity can be an indication of synonyms; however, this is not a necessary case for an extremely short distance. To a certain degree, the **CS** of NLIKR is similar to a lightweight universal ontology, but they are significantly different in terms of the purpose and information inclusion.

With millions of concepts, where each may possess millions of relations, the implementation of **CS** is an enormous task. Like the pixel representation of images, human knowledge representation should also start from the most fundamental element—relation. It may take several megabytes to represent one concept. Only in this way are we able to sufficiently and accurately describe each concept in a language that can be understood by a machine.

CONFLICT OF INTEREST

The Author declare no conflicts of interest.

AUTHORS' CONTRIBUTIONS

All researchers who contributed to the research are included as an author.

Funding Statement

The research is not funded by any funding source.

ACKNOWLEDGMENTS

One of the authors, Ms. Shirin Akther Khanam, a PhD research student, wishes to acknowledge the Postgraduate Research Scholarship awarded by La Trobe University (Australia), which financially supported her PhD study. "A Human-Machine Language Dictionary" is part of Ms. Khanam's PhD project.

REFERENCES

[1] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, K. Miller, WordNet: an on-line lexical database, Int. J. Lexicogr. 3 (2000), 235–244.
[2] H. Liu, P. Singh, ConceptNet - a practical commonsense reasoning tool kit, BT Technol. J. 22 (2004), 211–226.
[3] A. Carlson, J. Betteridge, B. Kisiel, *et al.*, Toward an architecture for never-ending language learning, in Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, 2010, pp. 1307–1313.
[4] W. Wilson, W. Liu, M. Bennamoun, Ontology learning from text: a look back and into the future, ACM Comput. Surv. 44 (2012), 1–36.
[5] J. Wang, H. Liu, H. Wang, A mapping-based tree similarity algorithm and its application to ontology alignment, Knowl. Based Syst. 56 (2014), 97–107.
[6] A. Solé-Ribalta, D. Sanchez, M. Batet, F. Serratosa, Towards the estimation of feature-based semantic similarity using multiple ontologies, Knowl. Based Syst. 55 (2014), 101–113.
[7] G. Li, F. Liu, Application of a clustering method on sentiment analysis, J. Inf. Sci. 38 (2012), 127–139.
[8] G. Li, F. Liu, Sentiment analysis based on clustering: a framework in improving accuracy and recognizing neutral opinions, Appl. Intell. 40 (2012), 441–452.
[9] F. Baader, Logic-based knowledge representation, in: M.J. Wooldridge, M. Veloso (Eds.), Artificial Intelligence Today, Lecture Notes in Computer Science, vol. 1600, Springer, Berlin, Heidelberg, 2001, pp. 13–41.
[10] P. Warren, P. Mulholland, T. Collins, E. Motta, Improving comprehension of knowledge representation languages: a case study with description logics, Int. J. Hum. Comput. Stud. 122 (2019), 145–167.
[11] R. Cyganiak, D. Wood, M. Lanthaler, RDF 1.1 concepts and abstract syntax, 2014. http://www.w3.org/TR/rdf11-concepts/
[12] P. Hitzler, OWL 2 Web Ontology Language Primer, second ed., 2014. http://www.w3.org/TR/2012/REC-owl2-primer-20121211/
[13] Y. Li, T. Li, J. Montero, Computational intelligence for emerging systems and applications, Int. J. Comput. Intell. Syst. 12 (2019), 474–475.
[14] D. Lenat, R. Guha, K. Oittman, D. Pratt, M. Shepherd, CYC: toward programs with common sense, Commun. ACM. 33 (1990), 30–49.
[15] B. Davis, Introduction to Lattices and Order, second ed., Cambridge University Press, Cambridge, UK, 1996.
[16] L. Stanchev, Creating a similarity graph from WordNet, in Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics, Thessaloniki, Greece, 2014.
[17] A. Evangelista, D. Kjos-Hanssen, Google distance between words, computing research repository, 2019. https://arxiv.org/pdf/0901.4180v1.pdf
[18] G. Yin, Q. Sheng, Research on ontology-based measuring semantic similarity, in Proceedings of the 2008 International Conference on Internet Computing in Science and Engineering, Harbin, China, 2008, pp. 250–253.
[19] S.A. Khanam, F. Liu, P.Y. Chen, Comprehensive structured knowledge base system construction with natural language presentation, Hum. Cent. Comput. Inf. Sci. 9 (2019).
[20] H. Dincer, S. Uzunkaya, S. Yuksel, An IT2-based hybrid decision-making model using hesitant fuzzy linguistics term sets for selecting the development plan of financial economics, Int. J. Comput. Intell. Syst. 12 (2019), 460–473.
[21] Y. Ou, L. Yi, B. Zou, Z. Pei, The linguistic intuitionistic fuzzy set TOPSIS method for linguistics multi-criteria decision makings, Int. J. Comput. Intell. Syst. 11 (2018), 120–132.