IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 8, 2018, accepted December 23, 2018, date of publication February 1, 2019, date of current version February 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891692

Challenging the Boundaries of Unsupervised Learning for Semantic Similarity

ATISH PAWAR AND VIJAY MAGO
Department of Computer Science, Lakehead University, Thunder Bay, ON P7B 5E1, Canada
Corresponding author: Atish Pawar (apawar1@lakeheadu.ca)

This work was supported by the Ontario Council on Articulation and Transfer (ONCAT) under Project 2017-17-LU.

ABSTRACT The semantic analysis field has a crucial role to play in the research related to text analytics. Calculating the semantic similarity between sentences is a long-standing problem in the area of natural language processing, and it differs significantly as the domain of operation differs. In this paper, we present a methodology that can be applied across multiple domains by incorporating corpora-based statistics into a standardized semantic similarity algorithm. To calculate the semantic similarity between words and sentences, the proposed method follows an edge-based approach using a lexical database. When tested on both benchmark standards and mean human similarity dataset, the methodology achieves a high correlation value for both word ($r = 0.8753$) and sentence similarity ($r = 0.8793$) concerning *Rubenstein and Goodenough* standard and the *SICK* dataset ($r = 0.8324$[1]) outperforming other unsupervised models.

INDEX TERMS Corpus, lexical database, natural language processing, semantic analysis, sentence similarity, word similarity.

I. INTRODUCTION

In general, semantic similarity is a measure of the conceptual distance between two objects, based on the correspondence of their meanings [1]. Semantic similarity between sentences in natural language processing (NLP) is considered a complex task, as the meaning of words changes significantly when the context is changed. As Jiang quotes, *"In many cases, humans have little difficulty in determining the intended meaning of an ambiguous word, while it is extremely difficult to replicate this process computationally"* [2]. Determination of semantic similarity in NLP has a wide range of applications. In internet-related applications, the uses of semantic similarity include estimating relatedness between search engine queries [3] and generating keywords for advertising on the web [4]. In biomedical applications, semantic similarity has become a valuable tool for analyzing results in gene clustering, gene expression and disease gene prioritization [5]–[7]. In addition to this, semantic similarity is also beneficial in information retrieval on web [8], text summarization [9] and text categorization [10]. Hence, such applications need to have a robust algorithm to estimate the semantic similarity which can be used across variety of domains.

Methodologies used to calculate semantic similarity are highly varied across multiple domains and the databases and algorithms used in one specific domain do not translate well onto other domains. Since the concept of calculating semantic similarities has a common underlying conceptual foundation regardless of domain, a methodology with a robust algorithm that can accurately estimate semantic similarity while incorporating a variety of domain specific predefined standard language measures is desirable. To improve the existing algorithms that determine the closeness of implications of the objects under comparison, it is clear that a domain specific predefined standard measure which readily describes the relatedness of the meanings in context is necessary. If we use natural language to compare the natural language sentences, then it would be a recursive problem with no stopping condition. Hence, it is essential to have some predefined measures.

This research aims to improve on existing algorithms and increase robustness through the integration of interchangeable domain specific corpora and through the use of lexical databases. Lexical databases have fixed vocabulary structures and the edge-based word structure that supports the determination of semantic similarity [11]. Many approaches utilizing lexical databases have been developed and proven to be very useful in the area of semantic analysis [2], [6], [12]–[15].

The main contribution of this research is a robust unsupervised semantic similarity algorithm which requires low computational resources and outperforms existing

[1]Eliminating the outliers which constitutes to 3.75% of 4927 statement pairs

algorithms relative to the Rubenstein and Goodenough(R&G) benchmark standard [16] and achieves a good correlation with respect to the SICK dataset [17].

The following section contains a review of related works. Section 3 provides a systematic review of our methodology. Section 4 explains the idea of traversal in a lexical database along with detailed visual diagrams and the computation with an illustrative example. Section 5 contains the result of our algorithm for the 65 noun word pairs from R&G [16] and sentence similarity for the sentence pairs in pilot data set [18] and and sentence similarity for the sentence pairs in SICK dataset [17]. Section 6 discusses the results and performance of the algorithm in relation to previous methodologies. Finally, section 7 briefly outlines the outcomes of this research with conclusions.

II. RELATED WORK

Recent work in the area of natural language processing has contributed valuable solutions to calculate the semantic similarity between words and sentences. This section reviews some related work to investigate the strengths and limitations of previous methods and to identify the particular difficulties in computing semantic similarity. Related works can roughly be classified into following major categories:

- Word co-occurrence methods
- Similarity based on a lexical database
- Method based on web search engine results
- Methods based on word vectors using recursive neural networks and deep neural networks

Word co-occurrence methods are commonly used in Information Retrieval (IR) systems [19]. This method has word list of meaningful words and every query is considered as a document. A vector is formed for the query and for documents. The relevant documents are retrieved based on the similarity between query vector and document vector [9]. This method has obvious drawbacks such as:

- It ignores the word order of the sentence.
- It does not take into account the meaning of the word in the context of the sentence.

But it has following advantages:

- It matches documents regardless the size of documents
- It successfully extracts keywords from documents [20]

Using the lexical database methodology, similarity is computed using a predefined word hierarchy which has words, meanings, and relationships with other words compiled in a tree-like structure [15]. While comparing two words, it takes into account the path distance between the words as well as the depth of the *subsumer* in the hierarchy. The subsumer refers to the relative root node concerning the two words being compared. It also uses a word corpus to calculate the 'information content'of the word which influences the final similarity. This methodology has the following limitations:

- The appropriate meaning of the word is not considered while calculating the similarity, rather it takes the best matching pair even if the meaning of the word is totally different in two distinct sentences.
- The information content of a word from one corpus differs from another.

The third methodology computes *relatedness* based on web search engine results utilizing the number of search results [21]. This technique does not necessarily give the similarity between words as words with opposite meanings frequently occur together on the web pages which influences the final similarity index. We have implemented the methodology to calcuate the Google Similarity Distance[2] [22]. The search engines that we used for this study are Google and Bing. The results obtained from this method are not encouraging.

Recently, the models based on neural networks have produced significant improvements in the results related to semantic similarity [23]–[27]. One revolutionary model proposed by Tai *et al.* (2015) [25] uses Glove vectors and subsequently Tree-LSTM. Tree-LSTMs generalize the order-sensitive chain-structure of standard LSTMs to tree-structured network topologies. A siamese adaptation of LSTM proposed by Mueller and Thyagarajan (2016) [24] outperforms the above mentioned neural networks based state of the art models. The authors explain the dependency of their model on a simple Manhattan metric. Their method forms a highly structured space whose geometry reflects complex semantic relationships. Performance evaluations for all aforementioned neural network models are trained on the SICK dataset [17] and tested on the same dataset. Despite improvements, these models perform poorly when tested on sentences which do not follow the grammar and structure of SICK sentences.

Overall, above-mentioned methods compute the semantic similarity without considering the context of the word according to the sentence. The algorithm proposed in this paper addresses aforementioned issues by disambiguating the words in sentences and forming semantic vectors dynamically for comparing sentences and words.

III. THE PROPOSED METHODOLOGY

The method[3] to calculate the semantic similarity between two sentences is divided into two modules:

Pass 1: Maximize the similarity
Pass 2: Bound the similarity

A. PASS 1: MAXIMIZE THE SIMILARITY

The proposed methodology considers the text as a sequence of words and deals with all the words in sentences separately according to their semantic and syntactic structure. The information content of the word is related to the frequency of the meaning of the word in a lexical database or a corpus. Figure 1 depicts the procedure to calculate the similarity between two sentences. Unlike other existing methods that use the fixed structure of vocabulary, the proposed method uses a lexical database to compare the appropriate meaning of the word. A semantic vector is formed for each sentence

[2]Interested readers can contact the authors for code and results.
[3]Algorithm is deployed at: http://www.loaga.science/algorithm

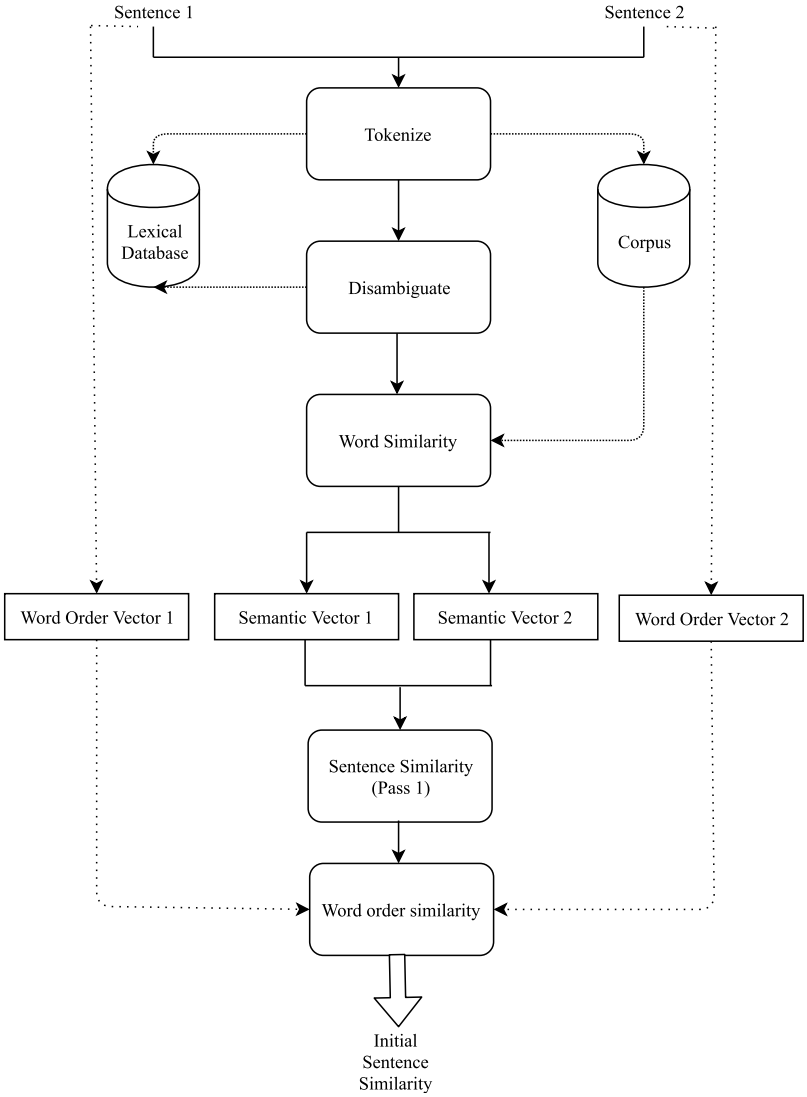

FIGURE 1. Proposed sentence similarity methodology. ***Tokenize***: POS tagging of words ***Disambiguate***: Identify appropriate synset ***Word Similarity***: Word similarity between two words ***Semantic vector***: Word similarities for words in sentences ***Word Order Vector:*** Occurences of words with respect to other sentence ***Sentence Similarity***: Intermediate sentence similarity and pass 2.

which contains the weight assigned to each word for every other word from the second sentence in comparison. This step also takes into account the information content of the word, for instance, word frequency from a standard corpus. Semantic similarity is calculated based on two semantic vectors. An order vector is formed for each sentence which considers the syntactic similarity between the sentences. Finally, semantic similarity is calculated based on semantic vectors and order vectors. *Pass 1* is divided into three parts:

- Word similarity
- Sentence similarity
- Word order similarity

The following section further describes each of the steps in more details.

1) WORD SIMILARITY

To compute the word similarity, the proposed method uses the sizeable lexical database for the English language, WordNet [28], from the Princeton University.

a: IDENTIFYING WORDS FOR COMPARISON

Before calculating the semantic similarity between words, it is essential to determine the words for comparison. We use word tokenizer and 'parts of speech tagging technique' as implemented in natural language processing toolkit, NLTK [29]. This step filters the input sentence and tags the words into their 'part of speech'(POS) and labels them accordingly. *WordNet* has path relationships between noun-noun and verb-verb only. Such relationships are absent in

WordNet for the other parts of speech. Hence, it is not possible to get a numerical value that represents the link between other parts of speech except nouns and verbs. We deal with other parts of speech in pass 2 of the algorithm.

Example: 'A voyage is a long journey on a ship or in a spacecraft'

TABLE 1. Parts of speeches.

Word	Part of Speech
A	DT - Determiner
voyage	NN - Noun
is	VBZ - Verb
a	DT - Determiner
long	JJ - Adjective
journey	NN - Noun
on	IN - Preposition
a	DT - Determiner
ship	NN - Noun
or	CC - Coordinating conjunction
in	IN - Prepostion
a	DT - Determiner
spacecraft	NN - Noun

Table 1 represents the words and the corresponding parts of speeches. The parts of speeches are as per the Penn Treebank [30].

b: ASSOCIATING WORD WITH A SENSE

The primary structure of WordNet is based on *synonymy*. Every word has *synsets* according to the meaning of the word in the context of a statement. The distance between *synsets* in comparison varies as we change the meaning of the word.

Consider an example where we calculate the shortest path distance between words 'river' and 'bank.' WordNet has only one synset for the word 'river'. We calculate the path distance between synset of 'river' and three synsets of word 'bank'. Table 2 represents the synsets and corresponding definitions for the words 'bank' and 'river'.

TABLE 2. Synsets and corresponding definitions from WordNet for words *bank* and *river*.

Synset	Definition
Synset('river.n.01')	a large natural stream of water (larger than a creek)
Synset('bank.n.01')	sloping land (especially the slope beside a body of water)
Synset('bank.n.09')	a building in which the business of banking transacted
Synset('bank.n.06')	the funds held by a gambling house or the dealer in some gambling games

Shortest distances for the Synset pairs are represented in Table 3. When comparing two sentences, we have many such word pairs which have multiple synsets. Therefore, not considering the proper synset in context of the sentence, could introduce errors at the early stage of similarity calculation.

TABLE 3. Synsets and corresponding shortest path distances from WordNet.

Synset Pair	Shortest Path Distance
Synset('river.n.01') - Synset('bank.n.01')	8
Synset('river.n.01') - Synset('bank.n.09')	10
Synset('river.n.01') - Synset('bank.n.06')	11

Hence, sense of the word has a significant effect on the overall similarity measure. Identifying the sense of the word is an area of research called 'word sense disambiguation'. We use 'max similarity' algorithm, Eq. 1, to perform word sense disambiguation [31] as implemented in Pywsd, an NLTK based Python library [32]. In Eq. 1, a is a query word and i represents all the words in context.

$$argmax_{synset(a)} = (\sum_{i}^{n} max_{synset(i)}(sim(i, a)) \tag{1}$$

c: SHORTEST PATH DISTANCE BETWEEN SYNSETS

Shortest path distance between synsets is the number of connecting edges between them in the lexical database, WordNet. The following example explains, in detail, the method used to calculate the shortest path distance. Referring to Figure 2, consider two words, viz.:

$w1 = motorcycle$ and $w2 = car$

We are referring to *Synset('motorcycle.n.01')* for 'motorcycle' and *('car.n.01')* for 'car'.

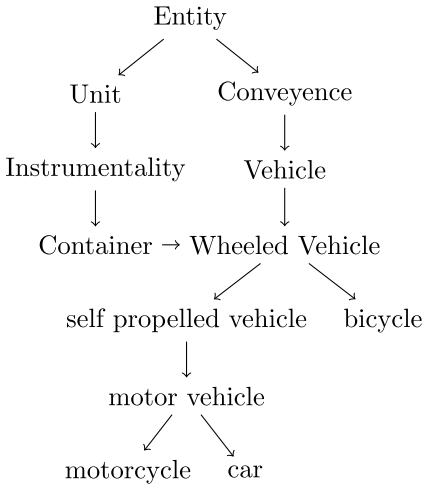

FIGURE 2. Hierarchical structure from WordNet.

The traversal path is: motorcycle → motor vehicle → car. Hence, the shortest path distance between motorcycle and car is 2. In WordNet, the gap between words increases as similarity decreases. Utilizing this property, we use the previously established monotonically decreasing function [15]:

$$f(l) = e^{-\alpha l} \tag{2}$$

where l is the shortest path distance and α is a constant. The selection of exponential function is to ensure that the value of *f(l)* lies between 0 to 1.

d: HIERARCHICAL DISTRIBUTION OF WORDS

In WordNet, the primary relationship between the synsets is the super-subordinate relation, also called *hyperonymy*, *hyponymy* or *ISA* relation [28]. This relationship connects the general concept synsets to the synsets that have specific characteristics. For example, Table 4 represents the word 'vehicle' and its *hyponyms*.

TABLE 4. Synset and corresponding hyponyms from WordNet.

Synset	Hyponyms
Synset('vehicle.n.01')	Synset('bumper_car.n.01') Synset('craft.n.02') Synset('military_vehicle.n.01') Synset('rocket.n.01') Synset('skibob.n.01') Synset('sled.n.01') Synset('steamroller.n.02') Synset('wheeled_vehicle.n.01')

The hyponyms of 'vehicle' have more specific properties and represent the particular set, whereas 'vehicle' has more general properties. Hence, words at the upper layer of the hierarchy have more general features and less semantic information, as compared to words at the lower layer of the hierarchy [15].

Hierarchical distance plays an important role when the path distances between word pairs are the same. For instance, referring to Figure 2, consider the following word pairs:

car - motorcycle and *bicycle - self_propelled_vehicle*.

The shortest path distance between both the pairs is 2, but the pair *car - motorcycle* has more semantic information and specific properties than *bicycle - self_propelled_vehicle*. Hence, we need to scale up the similarity measure if the word pair *subsume* words at the lower level of the hierarchy and scale down if they subsume words at the upper level of the hierarchy. To include this behavior, we use a previously established function [15]:

$$g(h) = \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \tag{3}$$

For WordNet, the optimal values of α and β are 0.2 and 0.45 respectively as reported previously [8].

2) INFORMATION CONTENT OF THE WORD

The meaning of the word differs as we change the domain of operation. We can use this behavior of natural language to make the similarity measure domain-specific. It is used to influence the similarity measure if the domain operation is predetermined. To illustrate the *Information Content* of the word in action, consider the word: *bank*. The most frequent meaning of the word *bank* in the context of Potamology (the study of rivers) is *sloping land (especially the slope beside a body of water)*. The most frequent meaning of the word *bank* in the context of Economics would be *a financial institution that accepts deposits and channels the money into lending activities*.

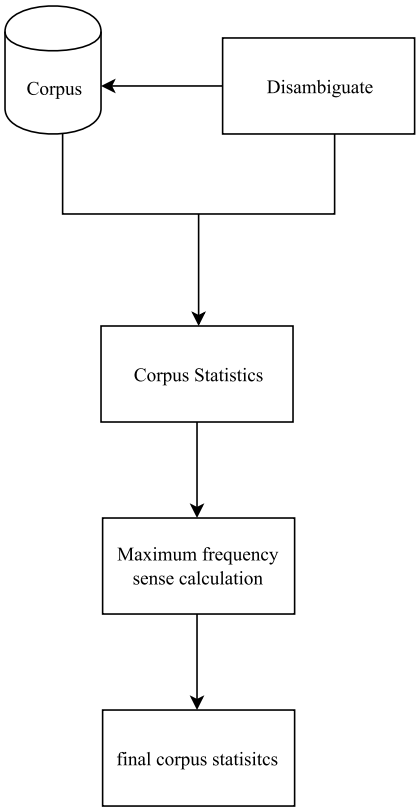

FIGURE 3. Method to calculate the frequency of a synset in a corpus. *Corpus*: An external corpus *Disambiguate*: Function to identify appropriate synset for every word in the corpus and write it in the corpus statistics file *Corpus Statistics*: An external file with corpus features *Maximum frequnecy sense calculation*: A function to determine the sysnet with maximum frequency for every word *Final corpus statistics*: An external file containing records from previous function

When applying the *Word Disambiguation Approach* described in section III-A.1.b, the final similarity of the word would be different for every corpus. The corpus, belonging to particular domain, works as supervised learning data for the algorithm. We first disambiguate the whole corpus to get the sense of the word and further calculate the frequency of the particular sense. These statistics for the corpus work as the *knowledge base* for the algorithm. Figure 3 represents the steps involved in the analysis of corpus statistics.

3) SENTENCES' SEMANTIC SIMILARITY

As Li *et al.* [15] states, the meaning of the sentence is reflected by the words in the sentence. Hence, we can use the semantic information from section III-A.1 and section III-A.2 to calculate the final similarity measure. Previously established methods to estimate the semantic similarity between sentences, use the static approaches like using a precompiled list of words and phrases. The problem with this technique is the precompiled list of words and phrases which doesn't necessarily reflect the correct semantic information in the context of compared sentences.

The dynamic approach includes the formation of a joint word vector which compiles words from sentences and uses

it as a baseline to form individual vectors. This method introduces inaccuracies in similarity calculations, particularly for the long sentences and the paragraphs containing multiple sentences.

Unlike these methods, our method forms the semantic value vectors for the sentences and aims to keep the size of the semantic value vector to the minimum. Formation of semantic vector begins after the section III-A.1.b. This approach avoids the overhead involved to form semantic vectors separately unlike done in previously discussed methods. Also, in this stage, we eliminate prepositions, conjunctions and interjections. Hence, these connectives are automatically eliminated from the semantic vector. We determine the size of the vector, based on the number of tokens from section III-A.1.b. Every unit of the semantic vector is initialized to null to void the foundational effect. Initializing the semantic vector to a unit positive value discards the negative/null effects, and overall semantic similarity will be a reflection of most similar words in the sentences. Let's see an example.

S1 = "A jewel is a precious stone used to decorate valuable things that you wear, such as rings or necklaces."

S2 = "A gem is a jewel or stone that is used in jewellery."

List of tagged words for *S1*:

[('jewel', Synset('jewel.n.01')), Synset('jewel.n.02')],
[('stone', Synset('stone.n.02')), Synset('stone.n.13')],
[('used', Synset('use.v.03')), Synset('use.v.06')],
[('decorate', Synset('decorate.v.01')), Synset('dress.v.09')],
[('valuable', Synset('valuable.a.01')), Synset('valuable.s.02')],
[('things', Synset('thing.n.04')), Synset('thing.n.12')],
[('wear', Synset('wear.v.01')), Synset('wear.v.09')],
[('rings', Synset('ring.n.08')), Synset('band.n.12')],
[('necklaces', Synset('necklace.n.01')), Synset('necklace.n.01')]

Length of the list of tagged words for *S1* is 9

List of tagged words for *S2*:

[('gem', Synset('jewel.n.01')), Synset('jewel.n.01')],
[('jewel', Synset('jewel.n.01')), Synset('jewel.n.02')],
[('stone', Synset('gem.n.02')), Synset('stone.n.13')],
[('used', Synset('use.v.03')), Synset('use.v.06')]
[('jewellery', Synset('jewelry.n.01')), Synset('jewelry.n.01')]

Length of the list of tagged words for *S2* is 5

We eliminate words like *a, is, to, that, you, such, as, or*; hence further reducing the computing overhead. The formed semantic vectors contain semantic information concerning all the words from both the sentences. For example, the semantic vector for *S1* is:

V1 = [0.99742103, 0.90118787, 0.42189901, 0.0, 0.0, 0.40630945, 0.0, 0.59202, 0.81750916]

Vector *V1* has semantic information from *S1* as well as from *S2*. Similarly, vector *V2* also has semantic information from *S1* and *S2*. To establish a similarity value using two vectors, we use the magnitude of the normalized vectors.

$$S = ||V1||.||V2|| \tag{4}$$

We make this method adaptable to longer sentences by introducing a variable(ζ) which is calculated dynamically at runtime. With the utilization of ζ, this method can also be used to compare paragraphs with multiple sentences.

a: DETERMINATION OF ζ

The words with maximum similarity have more impact on the magnitude of the vector. Using this property, we establish ζ for the sentences in comparison. According to R&G, the benchmark synonymy value of two words is 0.8025 [16]. Using this value as a determination standard, we calculate all the cells from *V1* and *V2* with the value greater than 0.8025. ζ is given by:

$$\zeta = sum(C1, C2)/\gamma \tag{5}$$

where $C1$ is count of valid elements in *V1* and $C2$ is count of valid elements in *V2*. γ is set to 1.8, determined by grid search over the correlation with R&G [16] and SICK dataset [17]. Now, using Eq. 4 and Eq. 7, we establish similarity as:

$$\delta = S/\zeta \tag{6}$$

b: DETERMINATION OF ζ

The words with maximum similarity have more impact on the magnitude of the vector. Using this property, we establish ζ for the sentences in comparison. According to R&G, the benchmark synonymy value of two words is 0.8025 [16]. Using this value as a determination standard, we calculate all the cells from *V1* and *V2* with the value greater than 0.8025. ζ is given by:

$$\zeta = sum(C1, C2)/\gamma \tag{7}$$

where $C1$ is count of valid elements in *V1* and $C2$ is count of valid elements in *V2*. γ is set to 1.8, determined by grid search over the correlation with R&G. Now, using Eq. 4 and Eq. 7, we establish similarity as:

$$\delta = S/\zeta \tag{8}$$

Algorithm 1 renders the explained procedure.

4) WORD ORDER SIMILARITY

Along with the semantic nature of the sentences, we need to consider the word order in the sentences. The word order similarity, simply put, is the aggregation of comparisons of word indices in two sentences. The semantic similarity approach based on words and the lexical database doesn't take into account the grammar of the sentence. Li *et al.* [15] assigns a number to each word in the sentence and forms a word order vector according to their occurrence and similarity. They also consider the semantic similarity value of words to decide the word order vector. If a word from sentence 1 is not present in sentence 2, the number assigned to the index

Algorithm 1 Semantic Similarity Between Sentences

```
procedure Sentence_similarity
    S1 − listoftaggedtokens ← disambiguate
    S2 − listoftaggedtokens ← disambiguate
    vector_length ← max(length(S1), length(S2))
    V1, V2 ← vector_length(null)
    V1, V2 ← vector_length(word_similarity(S1, S2))
    ζ = 0
    while S1_list_of _tagged_tokens do
        if word_similarity_value > benchmark_similarity_value then
            C1 ← C1 + 1
    while S2_list_of _tagged_tokens do
        if word_similarity_value > benchmark_similarity_value then
            C2 ← C2 + 1
    ζ ← sum(C1, C2)/γ
    S ← ||V1||.||V2||
    if sum(C1, C2) = 0 then
        ζ ← vector_length/2
    δ ← S/ζ
```

of this word in the word order vector corresponds to the word with maximum similarity. This case is not always valid and introduces errors in the final semantic similarity index. For the methods which calculate the similarity by chunking the sentence into words, it is not always necessary to decide the word order similarity. For such techniques, the word order similarity actually matters when two sentences contain same words in different order. Otherwise, if the sentences contain different words, the word order similarity should be an optional construct. In the entirely different sentences, word order similarity doesn't impact on the large scale. For such sentences, the impact of word order similarity is negligible as compared to the semantic similarity. Hence, in our approach, we implement word order similarity as an optional feature. Consider following classical example:

- *S1*: A quick brown dog jumps over the lazy fox.
- *S2*: A quick brown fox jumps over the lazy dog.

The edge-based approach using lexical database will produce a result showing that both *S1* and *S2* are same, but since the words appear in a different order we should scale down the overall similarity as they represent different meaning. We start with the formation of vectors *V1* and *V2* dynamically for sentences *S1* and *S2* respectively. Initialization of vectors is performed as explained in section III-A.3. Instead of forming joint word set, we treat sentences relatively to keep the size of vector to the minimum.

The process starts with the sentence having maximum length. Vector *V1* is formed with respect to sentence 1 and cells in *V1* are initialized to index values of words in *S1* beginning with 1. Hence *V1* for *S1* is:

$$V1 = [1, 2, 3, 4, 5, 6, 7, 8, 9]$$

Now, we form *V2* concerning *S1* and *S2*. To form *V2*, every word from *S2* is compared with *S1*. If the word from *S2* is absent in *S1*, then the cell in *V2* is filled with the index value of the word in sentence *S2*. If the word from *S2* matches with a word from *S1*, then the index of the word from *S1* is filled in *V2*.

In the above example, consider words 'fox' and 'dog' from sentence 2. The word 'fox' from *S2* is present in *S1* at the index 9. Hence, entry for 'fox' in *V2* would be 9. Similarly, the word 'dog' form *S2* is present in the *S1* at the index 4. Hence, entry for 'dog' in *V2* would be 4. Following the same procedure for all the words, we get *V2* as:

$$V2 = [1, 2, 3, 9, 5, 6, 7, 8, 4]$$

Finally, word order similarity is given by:

$$W_s = ||V1 - V2||/||V1 * V2|| \tag{9}$$

In this case, W_s is 0.067091.

B. PASS 2: BOUND THE SIMILARITY

The first pass of the algorithm returns the maximized similarity(δ) between two sentences. The second pass of the algorithm aims at computing a more robust similarity by reducing the ancillary similarity which causes skeweness in results by considering syntactical structure, adjectives and adverbs, and negations in the sentences. Skewness in this context implies the deviation of the similarity(δ) from the similarity in the SICK dataset.

We propose three approaches for the *Pass 2* of the algorithm.

1) Recurrence of words
2) Negation and stanford POS tagger model
3) Spacy's dependency parser model

1) MODEL 1: RECURRENCE OF WORDS

We consider the number of occurrences of a word with same meaning in the sentence. If a word occurs multiple times in the sentence, then we should reduce the impact of the word on the overall similarity. To illustrate this property of occurrences, consider following example:

S1: Explain the term Database and Database Management System DBMS, as well as the use of Primary and Foreign Key.

S2: Understand the fundamental concepts of relational database and implement a relational database.

The word *Database* occurs twice in both sentences. The impact it has on the final similarity is more than the actual information it adds to the sentence. Hence, while assigning the similarity value for such word pairs, we divide the subsequent occurrences by the number of occurrences.

$$V[word] = similarity/number\_of\_occurrences \tag{10}$$

where V represents the semantic vector. In this example, the value of similarity for *database* would be reduced to half as it occurs twice.

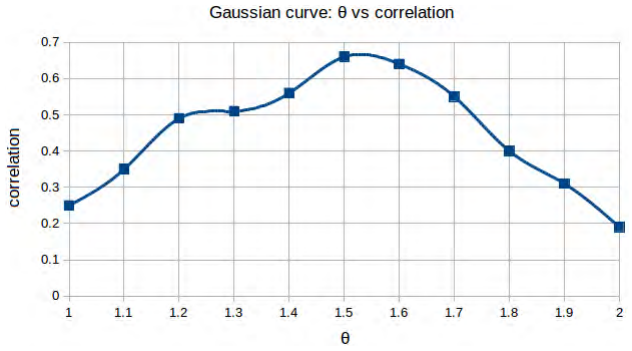

FIGURE 4. Normal distribution of θ over correlation.

2) MODEL 2: NEGATION AND STANFORD POS TAGGER MODEL

The intuitive idea behind this model is to build a concise list containing syntactical information for both sentences and subsequently processing the lists to arrive at a decision value. We focus on verbs, adverbs, and adjectives primarily. In this model, we use Stanford POS tagger, thesaurus.com Python API [33] and a list of English language contractions from Wikipedia [34]. We start by resolving the contractions to get the necessary form of the sentences. Both the sentences are tagged in their respective parts of speeches using Stanford's *bidirectional distsim tagger* [35]. A list is formed for both the sentences in following order:

1) The length of lists is determined by the length of the list containing POS of the sentences.

$$l = max(s1\_tagged, s2\_tagged)$$

2) All the elements in the list are initialized to zero.
3) If the word is verb, adverb, adjective or negation, then the corresponding bit is set to represent the POS of the word.

Both the lists are compared as depicted in Figure 5. A decision is made explicitly for each verb, adverb, and adjective. If opposite sense is encountered in the sentences, then similarity δ is amended using following formula:

$$\omega = \delta/\theta \tag{11}$$

θ is set to 1.5. Through grid search we found that θ at 1.5 gives the highest correlation with the SICK values. Figure 4 represents the normal distribution using Gaussian curve of correlation with respect to θ.

a: SPACY'S DEPENDENCY PARSER MODEL

The model based on dependency parsing outperforms model 1(III-B.1) and model 2(III-B.2). We use Spacy's [36] dependency parser to get the dependency grammar of the sentence. We follow a similar approach as in model 2 by forming a list representing the dependency information of the sentence. We assemble the following information from dependency parsing:

$$cell = \{token, token.pos, token.dep\}$$

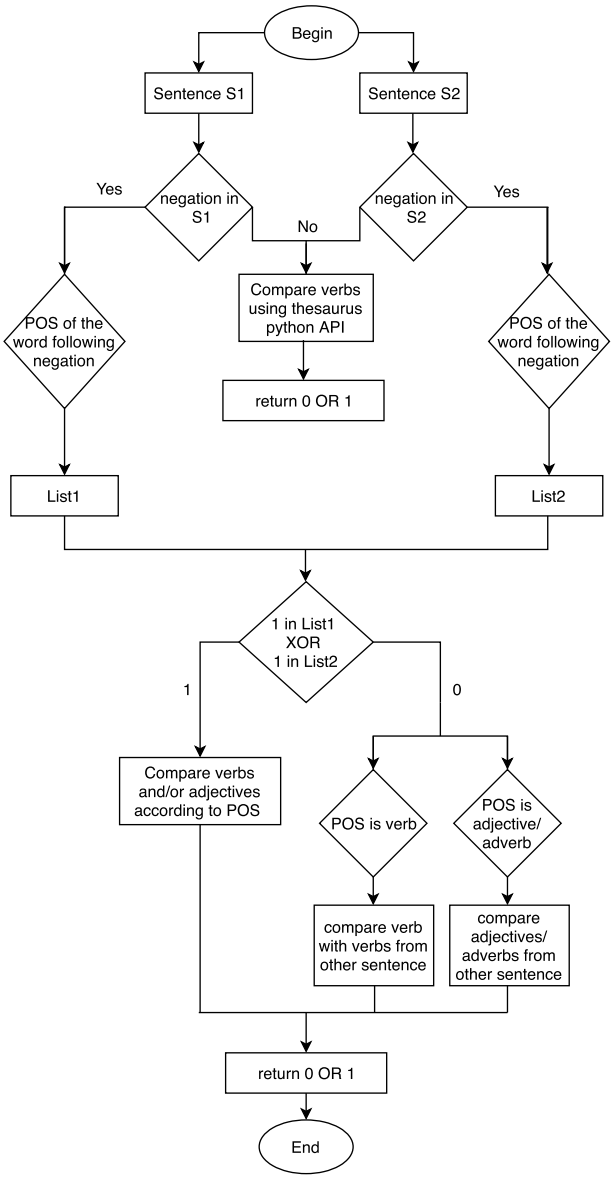

FIGURE 5. Decision making for negation 1 signifies the negation and 0 signifies no negation.

The above cell format represents a cell in the list. A token is a word from a sentence, token.pos is the part of speech of the token in a sentence, token.dep depicts the dependency in the sentence. We maintain information about root, nouns, and verbs from both the sentences separately.

The goal of this approach is to keep track of the syntactical differences by incrementing a global dependency variable. We start the comparison with the roots of both the sentences. If roots are not similar or if the synsets of roots do not intersect each other, then we increment the dependency variable by 1. Next, we compare the lists containing nouns and accordingly increment the dependency variable. We consider the length of the lists containing nouns and the dependency of the nouns in the sentence. Similarly, we compare the lists containing verbs.

We check the negation explicitly. We maintain a list of words conveying negation. We use the SICK dataset to compile this list. If we encounter a word from the list of negation words, then we increase the dependency variable(*dep_var*) by 1. Length of sentences is also an important factor affecting the semantics of the sentences. We use the following formula to calculate the *shift* between two sentences. Following formulae are derived considering the normal distribution of semantic similarity over SICK dataset [17].

$$shift = \epsilon * log(abs(s1\_length - s2\_length) + 1) \quad (12)$$

We establish a dependency index(*dep_index*) using following formula:

$$dep\_index = (\epsilon * tan^{-1}(dep\_var)) + shift \quad (13)$$

ϵ is set to 0.10 through grid search over correlation on SICK dataset. Finally, we use this dependency_index as a measure indicating the syntactical difference between two sentences. We establish final similarity as:

$$\omega = \delta - dep\_index \quad (14)$$

IV. IMPLEMENTATION USING SEMANTIC NETS

The database used to implement the proposed methodology is WordNet and statistical information from WordNet is used calculate the information content of the word. This section describes the prerequisites to implement the method.

A. THE DATABASE - WORDNET

WordNet is a lexical semantic dictionary available for online and offline use, developed and hosted at Princeton. The version used in this study is WordNet 3.0 which has 117,000 synonymous sets, *Synsets*. Synsets for a word represent the possible meanings of the word when used in a sentence. WordNet currently has synset structure for nouns, verbs, adjectives and adverbs. These lexicons are grouped separately and do not have interconnections; for instance, nouns and verbs are not interlinked.

The main relationship connecting the synsets is the super-subordinate(ISA-HASA) relationship. The relation becomes more general as we move up the hierarchy. The root node of all the noun hierarchies is 'Entity'. Like nouns, verbs are arranged into hierarchies as well.

1) SHORTEST PATH DISTANCE AND HIERARCHICAL DISTANCES FROM WORDNET

The WordNet relations connect the same parts of speeches. Thus, it consists of four subnets of nouns, verbs, adjectives and adverbs respectively. Hence, determining the similarity between cross-domains is not possible.

The shortest path distance is calculated by using the tree-like hierarchical structure. To figure the shortest path, we climb up the hierarchy from both the synsets and determine the meeting point which is also a synset. This synset is called *subsumer* of the respective synsets. The shortest path distance equals the tohops from one synset to another.

We consider the position of subsumer of two synsets to determine the hierarchical distance. Subsumer is found by using the *hyperonymy* (ISA) relation for both the synsets. The algorithm moves up the hierarchy until a common synset is found. This common synset is the subsumer for the synsets in comparison. A set of hypernyms is formed individually for each synset and the intersection of sets contains the subsumer. If the intersection of these sets contain more than one synset, then the synset with the shortest path distance is considered as a subsumer.

2) THE INFORMATION CONTENT OF THE WORD

For general purposes, we use the statistical information from WordNet for the information content of the word. WordNet provides the frequency of each synset in the WordNet corpus. This frequency distribution is used in the implementation of section III-A.2.

B. ILLUSTRATIVE EXAMPLE

This section explains in detail the steps involved in the calculation of semantic similarity between two sentences.

- *S1*: A gem is a jewel or stone that is used in jewellery.
- *S2*: A jewel is a precious stone used to decorate valuable things that you wear, such as rings or necklaces.

Following segment contains the parts of speeches and corresponding synsets used to determine the similarity.

For *S1* the tagged words are:

Synset('jewel.n.01'): a precious or semiprecious stone incorporated into a piece of jewelry

Synset('jewel.n.01'): a precious or semiprecious stone incorporated into a piece of jewelry

Synset('gem.n.02'): a crystalline rock that can be cut and polished for jewelry

Synset('use.v.03'): use up, consume fully

Synset('jewelry.n.01'): an adornment (as a bracelet or ring or necklace) made of precious metals and set with gems (or imitation gems)

For *S2* the tagged words are:

Synset('jewel.n.01'): a precious or semiprecious stone incorporated into a piece of jewelry

Synset('stone.n.02'): building material consisting of a piece of rock hewn in a definite shape for a special purpose

Synset('use.v.03'): use up, consume fully

Synset('decorate.v.01'): make more attractive by adding ornament, colour, etc.

Synset('valuable.a.01'): having great material or monetary value especially for use or exchange

Synset('thing.n.04'): an artifact

Synset('wear.v.01'): be dressed in

Synset('ring.n.08'): jewelry consisting of a circlet of precious metal (often set with jewels) worn on the finger

Synset('necklace.n.01'): jewelry consisting of a cord or chain (often bearing gems) worn about the neck as an ornament (especially by women)

TABLE 5. L1 compared with L2.

Words	Similarity
gem - jewel	**0.908008550956**
gem - stone	0.180732071642
gem - used	0.0
gem - decorate	0.0
gem - valuable	0.0
gem - things	0.284462910289
gem - wear	0.0
gem - rings	0.485032351325
gem - necklaces	0.669319889871
jewel - jewel	**0.997421032224**
jewel - stone	0.217431543606
jewel - used	0.0
jewel - decorate	0.0
jewel - valuable	0.0
jewel - things	0.406309448212
jewel - wear	0.0
jewel - rings	0.456849659596
jewel - necklaces	0.41718607131
stone - jewel	0.475813717007
stone - stone	**0.901187866267**
stone - used	0.0
stone - decorate	0.0
stone - valuable	0.0
stone - things	0.198770510639
stone - wear	0.0
stone - rings	0.100270000776
stone - necklaces	0.0856785820827
used - jewel	0.0
used - stone	0.0
used - used	**0.42189900525**
used - decorate	0.0
used - valuable	0.0
used - things	0.0
used - wear	0.0
used - rings	0.0
used - necklaces	0.0
jewellery - jewel	0.509332774797
jewellery - stone	0.220266070205
jewellery - used	0.0
jewellery - decorate	0.0
jewellery - valuable	0.0
jewellery - things	0.346687374295
jewellery - wear	0.0
jewellery - rings	0.592019999822
jewellery - necklaces	**0.81750915958**

TABLE 6. L2 compared with L1.

Words	Similarity
jewel - gem	0.908008550956
jewel - jewel	**0.997421032224**
jewel - stone	0.475813717007
jewel - used	0.0
jewel - jewellery	0.509332774797
stone - gem	0.180732071642
stone - jewel	0.217431543606
stone - stone	**0.901187866267**
stone - used	0.0
stone - jewellery	0.220266070205
used - gem	0.0
used - jewel	0.0
used - stone	0.0
used - used	**0.42189900525**
used - jewellery	0.0
decorate - gem	0.0
decorate - jewel	0.0
decorate - stone	0.0
decorate - used	0.0
decorate - jewellery	0.0
valuable - gem	0.0
valuable - jewel	0.0
valuable - stone	0.0
valuable - used	0.0
valuable - jewellery	0.0
things - gem	0.284462910289
things - jewel	**0.406309448212**
things - stone	0.198770510639
things - used	0.0
things - jewellery	0.346687374295
wear - gem	0.0
wear - jewel	0.0
wear - stone	0.0
wear - used	0.0
wear - jewellery	0.0
rings - gem	**0.485032351325**
rings - jewel	0.456849659596
rings - stone	0.100270000776
rings - used	0.0
rings - jewellery	0.592019999822
necklaces - gem	0.669319889871
necklaces - jewel	0.41718607131
necklaces - stone	0.0856785820827
necklaces - used	0.0
necklaces - jewellery	**0.81750915958**

After identifying the synsets for comparison, we find the shortest path distances between all the synsets and take the best matching result to form the semantic vector. The intermediate list is formed which contains the words and the identified synsets. L1 and L2 below represent the intermediate lists.

L1: [('gem', Synset('jewel.n.01'))], [('jewel', Synset('jewel.n.01'))], [('stone', Synset('gem.n.02'))], [('used', Synset('use.v.03'))], [('jewellery', Synset('jewelry.n.01'))]

L2: [('jewel', Synset('jewel.n.01'))], [('stone', Synset('stone.n.02'))], [('used', Synset('use.v.03'))], [('decorate', Synset('decorate.v.01'))], [('valuable', Synset('valuable.a.01'))], [('things', Synset('thing.n.04'))], [('wear', Synset('wear.v.01'))], [('rings', Synset('ring.n.08'))], [('necklaces', Synset('necklace.n.01'))]

Now we begin to form the semantic vectors for *S1* and *S2* by comparing every synset from *L1* with every synset from *L2*. The intermediate step here is to determine the size of semantic vector and initialize it to null. In this example, the size of the semantic vector is 9 by referring to the method explained in section III-A.3. The following part contains the cross comparison of *L1* and *L2*.

Cross-comparison with all the words from *S1* and *S2* is essential because if a word from statement *S1* best matches with a word from *S2*, it does not necessarily mean that it would be true if the case is reversed. This scenario can be observed with the words *jewel* from Table 5 and *things* from Table 6. *things* best matches with *jewel* with index

TABLE 7. Linear regression parameter values for proposed methodology.

Slope	0.84312603549362108
Intercept	0.017742354112473213
r-value	0.87536955005374539
p-value	1.4816200698817255e-21
stderr	0.058665976202757132

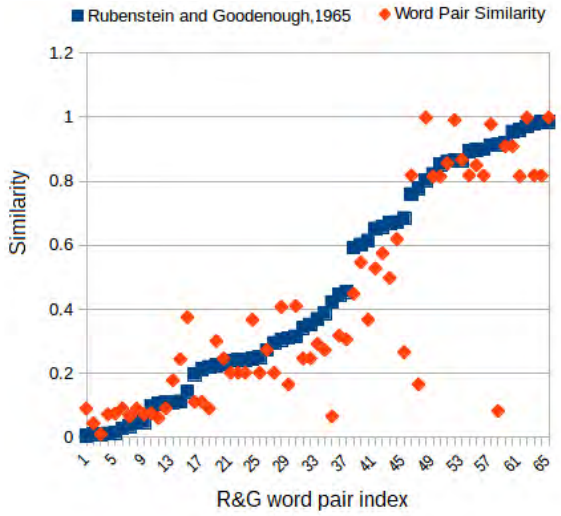

FIGURE 6. Perfomance of word similarity method vs Standard by Rubenstein and Goodenough.

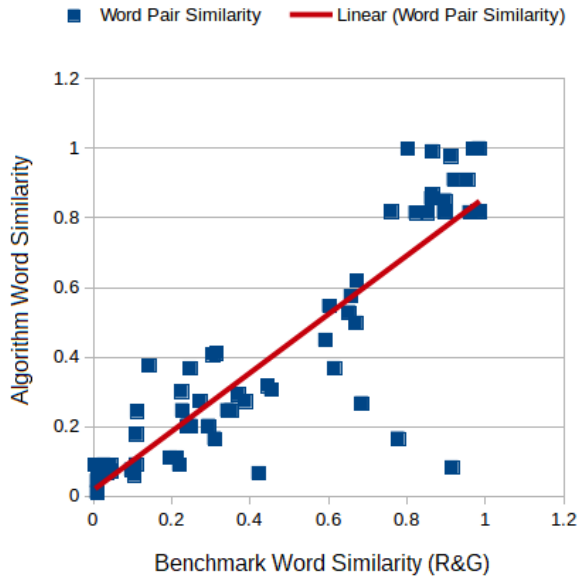

FIGURE 7. Linear Regression model word similarity method against Standard by Rubenstein and Goodenough.

of 0.4063 whereas *jewel* from Table 5 best matches with *jewel* from Table 6.

After getting the similarity values for all the word pairs, we need to determine an index entry for the semantic vector. The entry in the semantic vector for a word is the highest similarity value from the comparison with the words from

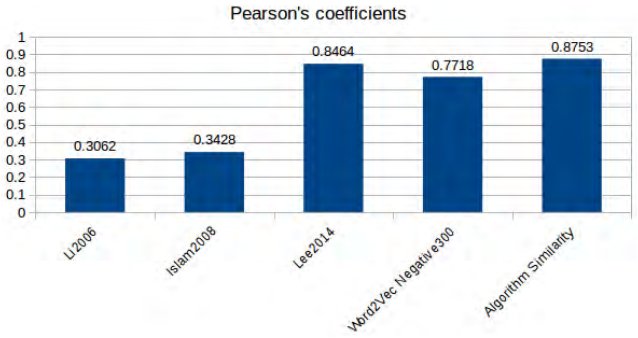

FIGURE 8. Pearson's coefficients from various algorithms against Standard by Rubenstein and Goodenough.

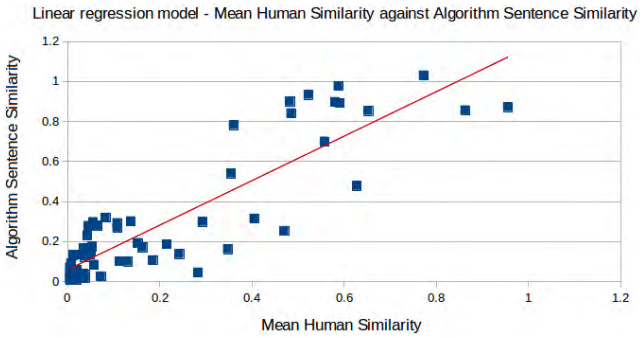

FIGURE 9. Linear regression model- mean human similarity vs algorithm sentence similarity.

other sentence. For instance, for the word *gem*, from Table 5, the corresponding semantic vector entry is 0.90800855 as it is the maximum of all the compared similarity values.

Hence, we get *V1* and *V2* as following:

- $V1$ = [0.90800855, 0.99742103, 0.90118787, 0.42189901, 0.81750916, 0.0, 0.0, 0.0, 0.0]
- $V2$ = [0.99742103, 0.90118787, 0.42189901, 0.0, 0.0, 0.40630945, 0.0, 0.59202, 0.81750916]

The intermediate step here is to calculate the dot product of the magnitude of normalized vectors: *V1* and *V2* as explained in section III-A.3.

$$S = 3.472426$$

The following segment explains the determination of ζ with reference to section III-A.3.b.

From Algorithm 1, *C1* for *V1* is 4. *C2* for *V2* is 3. Hence, ζ is (4+3)/1.8 = 3.89.

Now, the final similarity is

$\delta = S/\zeta = 3.472426/3.89 = 0.8929$.

We execute Pass 2 of the algorithm using dependency parser model. Here *length_difference* for *S*1 and *S*2 is 7. Hence we obtain a *shift* of 0.2079. Next the *dependency_variable* computed is 5 considering roots, negation if any, count and index of nouns and verbs in both the sentences. We obtain *dependency_index* of 0.1373.

Finally, $\delta = 0.8929-(0.2079+0.1373) = 0.5477$

TABLE 8. Results on the SICK semantic relatedness subtask. For our experiments, we report correlations and MSEs for 3 different models. Results are grouped as (1) previously reported supervised models (2) proposed unsupervised models.

Method	Pearson's r	Spearman's ρ	MSE
Supervised methods			
Illinois-LH (Lai and Hockenmaier, 2014) [26]	0.7993	0.7538	0.3692
UNAL-NLP (Jimenez et al., 2014) [39]	0.8070	0.7489	0.3550
Meaning Factory (Bjerva et al., 2014) [27]	0.8268	0.7721	0.3224
ECNU (Zhao et al., 2014) [40]	0.8414	-	-
Constituency Tree-LSTM(Kai et al., 2015) [25]	0.8582	0.7966	0.2734
Dependency Tree-LSTM(Kai et al., 2015) [25]	0.8676	0.8083	0.2532
ConvNet(he2015multi) [41]	0.8686	0.8047	0.2606
MaLSTM(Mueller et al., 2016) [24]	0.8822	0.8345	0.2286
Proposed unsupervised models			
Recurrence of words	0.5878	0.5147	0.5585
Negation and stanford POS tagger model	0.6645	0.5964	4389
Spacy's dependency parser model	**0.7958 (0.8324)**	**0.6854**	**0.3784**

V. EXPERIMENTAL RESULTS

To evaluate the algorithm, we used a three standard datasets:

- Rubenstein and Goodenough word pairs [16]
- Sentence similarity for Rubenstein and Goodenough word pairs [18]
- SICK test dataset [17]

The data has been used in many investigations over the years and has been established as a stable source of the semantic similarity measure. The word similarity obtained in this experiment is assisted by the standard sentences in Pilot Short Text Semantic Similarity Benchmark Data Set by O'Shea *et al.* [18]. The aim of this methodology is to achieve results as close as possible to the benchmark standards [16], [17]. The definitions of the words are obtained from the Collins Cobuild dictionary [37]. Our algorithm achieved a good Pearson correlation coefficient of 0.8753695501 for word similarity which is cosiderably higher than the existing algorithms.

Figure 6 shows the word pair similarity obtained from the algorithm along with the R&G similarity. Figure 8 represents the correlation results for 65 pairs from various algorithms against the R&G benchmark standard. Figure 9 represents the linear regression against the R&G standard. Table 7 shows the values of parameters for linear regression for word similarity and Figure 7 shows the corresponding linear regression. Proposed method outperforms all the existing methods concerning R&G benchmark standard for both word and sentence similarity.

A. SENTENCE SIMILARITY: R&G

Tables 11, 12 and 13 (see Appendix) contain the mean human sentence similarity values from Pilot Short Text Semantic Similarity Benchmark Data Set by O'Shea *et al.* [18]. As Li *et al.* [15] explains, when a survey was conducted by 32 participants to establish a measure for semantic similarity, they were asked to mark the sentences, not the words. Hence, word similarity is compared with the R&G [16] whereas sentence similarity is compared with mean human similarity. Our algorithm's sentence similarity achieved good Pearson correlation coefficient of 0.8794 with mean human similarity outperforming previous methods. Li *et al.* [15] obtained correlation coefficient of 0.816 and Islam and Inkpen [38] obtained correlation coefficient of 0.853. Out of 65 sentence pairs, 5 pairs were eliminated because of their definitions from Collins Cobuild dictionary [37]. The reasons and results are discussed in the discussion section.

B. SENTENCE SIMILARITY: SICK

To evaluate the sentence similarity algorithm, we used the SICK dataset which is considered as a stable measure of semantic correlation and has been used as a task in *SemEval 2014: semantic relatedness*. Our aim is to achieve semantic similarity as close as to the semantic similarity established in the SICK dataset. We present the results obtained from the three proposed models. Table 8 represents the correlations obtained for each model.

1) MODEL 1: RECURRENCE OF WORDS

Model 1 utilizes the property that the reoccurring words in the sentences contain less semantic information than the words occurring once. This property is useful when dealing with longer sentences. There are very few incidences in the SICK dataset which possess this property. We obtained a correlation of 0.58 concerning SICK dataset [17].

2) MODEL 2: NEGATION AND STANFORD POS TAGGER MODEL

We obtained a fairly good correlation of 0.66 for model 2 which uses Stanford POS tagger. This model performs well when all the words in both sentences are tagged correctly. It incurred few inaccuracies when negation is involved. The reason behind this behavior is the word following negation is tagged with a different POS than the corresponding word from the other sentence. Hence the negation calculation fails.

3) MODEL 3: SPACY'S DEPENDENCY PARSER MODEL

The dependency parser model performed best and obtained a correlation of 0.79 which is the best performing unsupervised model. We also encountered few *outliers*. *Outliers* are

TABLE 9. Rubenstein and Goodenough vs Lee 2014 vs proposed algorithm similarity.

R&GNo	R&Gpair	R&G Similarity [16]	Lee2014 [42]	GoogleNews English Negative300-Word2Vec [43]	Proposed Algorithm Similarity
1	cord smile	0.005	0.01	0.509055	0.0899021679
2	noon string	0.01	0.005	0.510825	0.0440401486
3	rooster voyage	0.01	0.0125	0.53135	0.010051669
4	fruit furnace	0.0125	0.0475	0.536605	0.0720444643
5	autograph shore	0.015	0.005	0.517325	0.0742552483
6	automobile wizard	0.0275	0.02	0.48596	0.0906955651
7	mound stove	0.035	0.005	0.6226	0.0656419906
8	grin implement	0.045	0.005	0.4999	0.0899021679
9	asylum fruit	0.0475	0.005	0.5289	0.0720444643
10	asylum monk	0.0975	0.0375	0.5693	0.0757289762
11	graveyard madhouse	0.105	0.0225	0.64695	0.0607950554
12	boy rooster	0.11	0.0075	0.6424	0.0907164485
13	glass magician	0.11	0.1075	0.51861	0.1782144411
14	cushion jewel	0.1125	0.0525	0.56235	0.2443794293
15	monk slave	0.1425	0.045	0.5957	0.3750880747
16	asylum cemetery	0.1975	0.0375	0.5462	0.1106378337
17	coast forest	0.2125	0.0475	0.6180485	0.1106378337
18	grin lad	0.22	0.0125	0.624005	0.0899021679
19	shore woodland	0.225	0.0825	0.55845	0.3011198804
20	monk oracle	0.2275	0.1125	0.65177	0.2464473057
21	boy sage	0.24	0.0425	0.582975	0.2017739882
22	automobile cushion	0.2425	0.02	0.56675	0.2018466921
23	mound shore	0.2425	0.035	0.56582	0.2018466921
24	lad wizard	0.2475	0.0325	0.665	0.3673305438
25	forest graveyard	0.25	0.065	0.614505	0.2015952767
26	food rooster	0.2725	0.055	0.55915	0.2732326922
27	cemetery woodland	0.295	0.0375	0.69096	0.2015952767
28	shore voyage	0.305	0.02	0.60215	0.4075214431
29	bird woodland	0.31	0.0125	0.670121	0.1651985693
30	coast hill	0.315	0.1	0.58055	0.4103617321
31	furnace implement	0.3425	0.05	0.5117	0.2464473057
32	crane rooster	0.3525	0.02	0.618035	0.2465928735
33	hill woodland	0.37	0.145	0.63675	0.2918421392
34	car journey	0.3875	0.0725	0.549245	0.2730713984
35	cemetery mound	0.4225	0.0575	0.60302	0.0656419906
36	glass jewel	0.445	0.1075	0.5872465	0.3176716099
37	magician oracle	0.455	0.13	0.6261	0.3057403627
38	crane implement	0.5925	0.185	0.51159	0.4486585394
39	brother lad	0.6025	0.1275	0.67975	0.5462290271
40	sage wizard	0.615	0.1525	0.669055	0.3675115617
41	oracle sage	0.6525	0.2825	0.72125	0.5279307332
42	bird cock	0.6575	0.035	0.681	0.5750838807
43	bird crane	0.67	0.1625	0.6514	0.4978503715
44	food fruit	0.6725	0.2425	0.687046	0.6196075053
45	brother monk	0.685	0.045	0.6116	0.2664571358
46	asylum madhouse	0.76	0.215	0.6262695	0.8185286992
47	furnace stove	0.7775	0.3475	0.80415	0.1651985693
48	magician wizard	0.8025	0.355	0.74315	0.9985079423
49	hill mound	0.8225	0.2925	0.7311	0.8148010746
50	cord string	0.8525	0.47	0.59475	0.8148010746
51	glass tumbler	0.8625	0.1375	0.733755	0.8561402541
52	grin smile	0.865	0.485	0.9302	0.9910074537
53	serf slave	0.865	0.4825	0.7249	0.8673305438
54	journey voyage	0.895	0.36	0.8415	0.8185286992
55	autograph signature	0.8975	0.405	0.6566	0.8499457067
56	coast shore	0.9	0.5875	0.75415	0.8179120223
57	forest woodland	0.9125	0.6275	0.82085	0.9780261147
58	implement tool	0.915	0.59	0.60617	0.0822919486
59	cock rooster	0.92	0.8625	0.7393	0.9093502924
60	boy lad	0.955	0.58	0.7943	0.9093502924
61	cushion pillow	0.96	0.5225	0.6258	0.8157293861
62	cemetery graveyard	0.97	0.7725	0.8212403	0.9985079423
63	automobile car	0.98	0.5575	0.791915	0.8185286992
64	gem jewel	0.985	0.955	0.8105	0.8175091596
65	midday noon	0.985	0.6525	0.77637	0.9993931059

TABLE 10. Proposed algorithm similarity vs Islam2008 vs Li2006.

R&G No	R&G pair [16]	Proposed Algorithm Similarity	A.Islam 2008 [38]	Li et al.2006 [15]
1	cord smile	0.0899021679	0.06	0.33
5	autograph shore	0.0742552483	0.11	0.29
9	asylum fruit	0.0720444643	0.07	0.21
12	boy rooster	0.0907164485	0.16	0.53
17	coast forest	0.1106378337	0.26	0.36
21	boy sage	0.2017739882	0.16	0.51
25	forest graveyard	0.2015952767	0.33	0.55
29	bird woodland	0.1651985693	0.12	0.33
33	hill woodland	0.2918421392	0.29	0.59
37	magician oracle	0.3057403627	0.2	0.44
41	oracle sage	0.5279307332	0.09	0.43
47	furnace stove	0.1651985693	0.3	0.72
48	magician wizard	0.9985079423	0.34	0.65
49	hill mound	0.8148010746	0.15	0.74
50	cord string	0.8148010746	0.49	0.68
51	glass tumbler	0.8561402541	0.28	0.65
52	grin smile	0.9910074537	0.32	0.49
53	serf slave	0.8673305438	0.44	0.39
54	journey voyage	0.8185286992	0.41	0.52
55	autograph signature	0.8499457067	0.19	0.55
56	coast shore	0.8179120223	0.47	0.76
57	forest woodland	0.9780261147	0.26	0.7
58	implement tool	0.0822919486	0.51	0.75
59	cock rooster	0.9093502924	0.94	1
60	boy lad	0.9093502924	0.6	0.66
61	cushion pillow	0.8157293861	0.29	0.66
62	cemetery graveyard	0.9985079423	0.51	0.73
63	automobile car	0.8185286992	0.52	0.64
64	gem jewel	0.8175091596	0.65	0.83
65	midday noon	0.9993931059	0.93	1

the cases where either the disambiguation function fails to identify the correct synset for the word or the dependency parser fails to form the fitting dependency model. Our algorithm's measure obtained the correlation of 0.83 by eliminating the outliers.

VI. DISCUSSION

Our algorithm's similarity measure achieved a good Pearson correlation coefficient of 0.8753 with R&G word pairs [16]. This performance outperforms all the previous methods. Table 9 represents the comparison of similarity from proposed method and Lee *et al.* [42] with the R&G. Table 10 depicts the comparison of algorithm similarity against Islam and Inkpen [38] and Li *et al.* [15] for the 30 noun pairs and performs better.

For sentence similarity, the pairs *17: coast-forest, 24: lad-wizard, 30: coast-hill, 33: hill-woodland and 39: brother-lad* are not considered. The reason for this is, the definition of these word pairs have more than one common or synonymous words. Hence, the overall sentence similarity does not reflect the true sense of these word pairs as they are rated with low similarity in mean human ratings. For example, the definition of 'lad' is given as: 'A lad is a young man or boy.' and the definition of 'wizard' is: 'In legends and fairy stories, a wizard is a man who has magic powers.' Both sentences have similar or closely related words such as: 'man-man', 'boy-man' and 'lad-man'. Hence, these pairs affect overall similarity measure more than the actual words compared 'lad-wizard'.

VII. CONCLUSIONS

This paper presented an unsupervised approach to calculate the semantic similarity between two words, sentences or paragraphs which is applicable across multiple domains. The ability to accurately predict semantic similarity using a robust algorithm, a standardized lexical database and interchangable corpora with low computing overhead is beneficial to professionals in all domains requiring semantic similarity calculations. The algorithm initially disambiguates both the sentences and tags them in their parts of speeches. The disambiguation approach ensures the right meaning of the word for comparison. The similarity between words is calculated based on a previously established edge-based approach. The information content from a corpus can be used to influence the similarity in particular domain. Semantic vectors containing similarities between words are formed for sentences and further used for sentence similarity calculation. Word order vectors are also formed to calculate the impact of the syntactic structure of the sentences. Since word order affects less on the overall similarity than that of semantic similarity, word order similarity is weighted to a smaller extent. The methodology has been tested on previously established data

TABLE 11. Sentence similarity from proposed methodology compared with human mean similarity from Li2006.

R&G number	Sentence 1	Sentence 2	Mean Human Similarity	Proposed Algorithm Sentence Similarity
1	Cord is strong, thick string.	A smile is the expression that you have on your face when you are pleased or amused, or when you are being friendly.	0.01	0.0125
2	A rooster is an adult male chicken.	A voyage is a long journey on a ship or in a spacecraft.	0.005	0.1593
3	Noon is 12 o'clock in the middle of the day.	String is thin rope made of twisted threads, used for tying things together or tying up parcels.	0.0125	0.03455
4	Fruit or a fruit is something which grows on a tree or bush and which contains seeds or a stone covered by a substance that you can eat.	A furnace is a container or enclosed space in which a very hot fire is made, for example to melt metal, burn rubbish or produce steam.	0.0475	0.1388
5	An autograph is the signature of someone famous which is specially written for a fan to keep.	The shores or shore of a sea, lake, or wide river is the land along the edge of it.	0.0050	0.0701
6	An automobile is a car.	In legends and fairy stories, a wizard is a man who has magic powers.	0.0200	0.0088
7	A mound of something is a large rounded pile of it.	A stove is a piece of equipment which provides heat, either for cooking or for heating a room.	0.0050	0.3968
8	A grin is a broad smile.	An implement is a tool or other pieces of equipment.	0.0050	0.0099
9	An asylum is a psychiatric hospital.	Fruit or a fruit is something which grows on a tree or bush and which contains seeds or a stone covered by a substance that you can eat.	0.0050	0.01456
10	An asylum is a psychiatric hospital.	A monk is a member of a male religious community that is usually separated from the outside world.	0.0375	0.0175
11	A graveyard is an area of land, sometimes near a church, where dead people are buried.	If you describe a place or situation as a madhouse,you mean that it is full of confusion and noise.	0.0225	0.1339
12	Glass is a hard transparent substance that is used to make things such as windows and bottles.	A magician is a person who entertains people by doing magic tricks.	0.0075	0.0911
13	A boy is a child who will grow up to be a man.	A rooster is an adult male chicken.	0.1075	0.2921
14	A cushion is a fabric case filled with soft material, which you put on a seat to make it more comfortable.	A jewel is a precious stone used to decorate valuable things that you wear, such as rings or necklaces.	0.0525	0.1745
15	A monk is a member of a male religious community that is usually separated from the outside world.	A slave is someone who is the property of another person and has to work for that person.	0.0450	0.1394
16	An asylum is a psychiatric hospital.	A cemetery is a place where dead peoples bodies or their ashes are buried.	0.375	0.03398
17	The coast is an area of land that is next to the sea.	A forest is a large area where trees grow close together.	0.0475	0.3658
18	A grin is a broad smile.	A lad is a young man or boy.	0.0125	0.0281
19	The shores or shore of a sea, lake, or wide river is the land along the edge of it.	Woodland is land with a lot of trees.	0.0825	0.2192
20	A monk is a member of a male religious community that is usually separated from the outside world.	In ancient times, an oracle was a priest or priestess who made statements about future events or about the truth.	0.1125	0.1011
21	A boy is a child who will grow up to be a man.	A sage is a person who is regarded as being very wise.	0.0425	0.2305
22	An automobile is a car.	A cushion is a fabric case filled with soft material, which you put on a seat to make it more comfortable.	0.0200	0.0330
23	A mound of something is a large rounded pile of it.	The shores or shore of a sea, lake, or wide river is the land along the edge of it.	0.0350	0.0386
24	A lad is a young man or boy.	In legends and fairy stories, a wizard is a man who has magic powers.	0.0325	0.2939
25	A forest is a large area where trees grow close together.	A graveyard is an area of land, sometimes near a church, where dead people are buried.	0.0650	0.2787
26	Food is what people and animals eat.	A rooster is an adult male chicken.	0.0550	0.2972
27	A cemetery is a place where dead peoples bodies or their ashes are buried.	Woodland is land with a lot of trees.	0.0375	0.1240
28	The shores or shore of a sea, lake, or wide river is the land along the edge of it.	A voyage is a long journey on a ship or in a spacecraft.	0.0200	0.0304
29	A bird is a creature with feathers and wings, females lay eggs, and most birds can fly.	Woodland is land with a lot of trees.	0.0125	0.1334
30	The coast is an area of land that is next to the sea.	A hill is an area of land that is higher than the land that surrounds it.	0.1000	0.8032
31	A furnace is a container or enclosed space in which a very hot fire is made, for example to melt metal, burn rubbish or produce steam.	An implement is a tool or other piece of equipment.	0.0500	0.1408
32	A crane is a large machine that moves heavy things by lifting them in the air.	A rooster is an adult male chicken.	0.0200	0.0564
33	A hill is an area of land that is higher than the land that surrounds it.	Woodland is land with a lot of trees.	0.1450	0.6619

sets which contain standard results as well as mean human results. Our algorithm achieved a good Pearson correlation coefficient of 0.8753 for word similarity concerning the benchmark standard and 0.8794 (Table 9) for sentence similarity with respect to mean human similarity (Table 11-13) and 0.7958 concerning the SICK dataset.

TABLE 12. Sentence similarity from proposed methodology compared with human mean similarity from li2006 (continued from previous page).

R&G Number	Sentence 1	Sentence 2	Mean Human Similarity	Proposed Algorithm Sentence Similarity
34	A car is a motor vehicle with room for a small number of passengers.	When you make a journey, you travel from one place to another.	0.0725	0.02610
35	A cemetery is a place where dead peoples bodies or their ashes are buried.	A mound of something is a large rounded pile of it.	0.0575	0.0842
36	Glass is a hard transparent substance that is used to make things such as windows and bottles.	A jewel is a precious stone used to decorate valuable things that you wear, such as rings or necklaces.	0.1075	0.2692
37	A magician is a person who entertains people by doing magic tricks.	In ancient times, an oracle was a priest or priestess who made statements about future events or about the truth.	0.1300	0.1000
38	A crane is a large machine that moves heavy things by lifting them in the air.	An implement is a tool or other piece of equipment.	0.1850	0.1060
39	Your brother is a boy or a man who has the same parents as you.	A lad is a young man or boy.	0.1275	0.8615
40	A sage is a person who is regarded as being very wise.	In legends and fairy stories, a wizard is a man who has magic powers.	0.1525	0.1920
41	In ancient times, an oracle was a priest or priestess who made statements about future events or about the truth.	A sage is a person who is regarded as being very wise.	0.2825	0.0452
42	A bird is a creature with feathers and wings, females lay eggs, and most birds can fly.	A crane is a large machine that moves heavy things by lifting them in the air.	0.0350	0.1660
43	A bird is a creature with feathers and wings, females lay eggs, and most birds can fly.	A cock is an adult male chicken.	0.1625	0.1704
44	Food is what people and animals eat.	Fruit or a fruit is something which grows on a tree or bush and which contains seeds or a stone covered by a substance that you can eat.	0.2425	0.1379
45	Your brother is a boy or a man who has the same parents as you.	A monk is a member of a male religious community that is usually separated from the outside world.	0.0450	0.2780
46	An asylum is a psychiatric hospital.	If you describe a place or situation as a madhouse, you mean that it is full of confusion and noise.	0.2150	0.1860
47	A furnace is a container or enclosed space in which a very hot fire is made, for example, to melt metal, burn rubbish, or produce steam.	A stove is a piece of equipment which provides heat, either for cooking or for heating a room.	0.3475	0.1613
48	A magician is a person who entertains people by doing magic tricks.	In legends and fairy stories, a wizard is a man who has magic powers.	0.3550	0.5399
49	A hill is an area of land that is higher than the land that surrounds it.	A mound of something is a large rounded pile of it.	0.2925	0.2986
50	Cord is strong, thick string.	String is thin rope made of twisted threads, used for tying things together or tying up parcels.	0.4700	0.2530
51	Glass is a hard transparent substance that is used to make things such as windows and bottles.	A tumbler is a drinking glass with straight sides.	0.1375	0.2643
52	A grin is a broad smile.	A smile is the expression that you have on your face when you are pleased or amused, or when you are being friendly.	0.4850	0.7204
53	In former times, serfs were a class of people who had to work on a particular persons land and could not leave without that persons permission.	A slave is someone who is the property of another person and has to work for that person.	0.4825	0.7695
54	When you make a journey, you travel from one place to another.	A voyage is a long journey on a ship or in a spacecraft.	0.3600	0.7201
55	An autograph is the signature of someone famous which is specially written for a fan to keep.	Your signature is your name, written in your own characteristic way, often at the end of a document to indicate that you wrote the document or that you agree with what it says.	0.4050	0.3146
56	The coast is an area of land that is next to the sea.	The shores or shore of a sea, lake, or wide river is the land along the edge of it.	0.5875	0.7945
57	A forest is a large area where trees grow close together.	Woodland is land with a lot of trees.	0.6275	0.4770
58	An implement is a tool or other pieces of equipment.	A tool is any instrument or simple piece of equipment that you hold in your hands and use to do a particular kind of work.	0.5900	0.7590
59	A cock is an adult male chicken.	A rooster is an adult male chicken.	0.8625	0.8560
60	A boy is a child who will grow up to be a man.	A lad is a young man or boy.	0.5800	0.8296
61	A cushion is a fabric case filled with soft material, which you put on a seat to make it more comfortable.	A pillow is a rectangular cushion which you rest your head on when you are in bed.	0.5225	0.7626
62	A cemetery is a place where dead peoples bodies or their ashes are buried.	A graveyard is an area of land, sometimes near a church, where dead people are buried.	0.7725	0.8750
63	An automobile is a car.	A car is a motor vehicle with room for a small number of passengers.	0.5575	0.7001
64	Midday is 12 oclock in the middle of the day.	Noon is 12 oclock in the middle of the day.	0.9550	0.8726

TABLE 13. Sentence similarity from proposed methodology compared with human mean similarity from Li2006 (continued from previous page).

R&G Number	Sentence 1	Sentence 2	Mean Human Similarity	Proposed Algorithm Sentence Similarity
65	A gem is a jewel or stone that is used in jewellery.	A jewel is a precious stone used to decorate valuable things that you wear, such as rings or necklaces.	0.6525	0.5477

TABLE 14. Sentence similarity from proposed methodology compared with SICK similarity.

SICK index	Sentence 1	Sentence 2	SICK similarity	Proposed algorithm similarity
6	There is no boy playing outdoors and there is no man smiling	A group of kids is playing in a yard and an old man is standing in the background	0.575	0.5054
7	A group of boys in a yard is playing and a man is standing in the background	The young boys are playing outdoors and the man is smiling nearby	0.675	0.6689
8	A group of children is playing in the house and there is no man standing in the background	The young boys are playing outdoors and the man is smiling nearby	0.5	0.5054
10	A brown dog is attacking another animal in front of the tall man in pants	A brown dog is attacking another animal in front of the man in pants	0.975	0.8892
11	A brown dog is attacking another animal in front of the man in pants	A brown dog is helping another animal in front of the man in pants	0.66625	0.5966
13	Two dogs are wrestling and hugging	There is no dog wrestling and hugging	0.575	0.6306
15	A brown dog is attacking another animal in front of the man in pants	There is no dog wrestling and hugging	0.425	0.4920
16	Two dogs are wrestling and hugging	A brown dog is attacking another animal in front of the tall man in pants	0.475	0.4950
17	Two dogs are wrestling and hugging	A brown dog is helping another animal in front of the man in pants	0.325	0.5034
19	A person in a black jacket is doing tricks on a motorbike	A man in a black jacket is doing tricks on a motorbike	0.975	0.95

APPENDIX A
TABLES

See Tables 9–14.

ACKNOWLEDGMENT

The authors would like to thank Salimur Choudhury for his insight on different aspects of this project; Danny Kivi for setting up the online demo;[4] Andrew Heppner and the datalab.science team for reviewing and proofreading the paper.

REFERENCES

[1] D. Lin, "An information-theoretic definition of similarity," in *Proc. ICML*, vol. 98. 1998, pp. 296–304.

[2] J. J. Jiang and D. W. Conrath. (1997). "Semantic similarity based on corpus statistics and lexical taxonomy." [Online]. Available: https://arxiv.org/abs/cmp-lg/9709008

[3] A. Freitas, J. G. Oliveira, S. O'Riain, E. Curry, and J. C. P. da Silva, "Querying linked data using semantic relatedness: A vocabulary independent approach," in *Proc. Int. Conf. Natural Lang. Process. Inf. Syst.*, 2011, pp. 40–51.

[4] V. Abhishek and K. Hosanagar, "Keyword generation for search engine advertising using semantic similarity between terms," in *Proc. 9th Int. Conf. Electron. Commerce*, 2007, pp. 89–94.

[5] C. Pesquita, D. Faria, A. O. Falcao, P. Lord, and F. M. Couto, "Semantic similarity in biomedical ontologies," *PLoS Comput. Biol.*, vol. 5, no. 7, p. e1000443, 2009.

[6] P. W. Lord, R. D. Stevens, A. Brass, and C. A. Goble, "Investigating semantic similarity measures across the Gene Ontology: The relationship between sequence and annotation," *Bioinformatics*, vol. 19, no. 10, pp. 1275–1283, 2003.

[7] T. Pedersen, S. V. Pakhomov, S. Patwardhan, and C. G. Chute, "Measures of semantic similarity and relatedness in the biomedical domain," *J. Biomed. Inform.*, vol. 40, no. 3, pp. 288–299, 2007.

[8] G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. Petrakis, and E. E. Milios, "Semantic similarity methods in wordNet and their application to information retrieval on the Web," in *Proc. 7th Annu. ACM Int. Workshop Web Inf. Data Manage.*, 2005, pp. 10–16.

[9] G. Erkan and D. R. Radev, "LexRank: Graph-based lexical centrality as salience in text summarization," *J. Artif. Intell. Res.*, vol. 22, pp. 457–479, Dec. 2004.

[10] Y. Ko, J. Park, and J. Seo, "Improving text categorization using the importance of sentences," *Inf. Process. Manage.*, vol. 40, no. 1, pp. 65–79, 2004.

[11] C. Fellbaum, "WordNet," in *Theory and Applications of Ontology: Computer Applications*. Hoboken, NJ, USA: Wiley, 1998.

[12] A. D. Baddeley, "Short-term memory for word sequences as a function of acoustic, semantic and formal similarity," *Quart. J. Exp. Psychol.*, vol. 18, no. 4, pp. 362–365, 1966.

[13] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *J. Artif. Intell. Res.*, vol. 11, pp. 95–130, Jul. 1999.

[14] G. A. Miller and W. G. Charles, "Contextual correlates of semantic similarity," *Lang. Cogn. Process.*, vol. 6, no. 1, pp. 1–28, 1991.

[15] Y. Li, D. McLean, Z. A. Bandar, and J. D. O'Shea, and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 8, pp. 1138–1150, Aug. 2006.

[16] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," *Commun. ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[17] M. Marelli *et al.*, "A SICK cure for the evaluation of compositional distributional semantic models," in *Proc. LREC*, 2014, pp. 216–223.

[18] J. O'Shea, Z. Bandar, K. Crockett, and D. McLean, "Pilot short text semantic similarity benchmark data set: Full listing and description," *Computing*, Jul. 2009.

[19] C. T. Meadow, *Text Information Retrieval Systems*. New York, NY, USA: Academic, 1992.

[20] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," *Int. J. Artif. Intell. Tools*, vol. 13, no. 1, pp. 157–169, 2004.

[4] http://www.loaga.science/algorithm

[21] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Measuring semantic similarity between words using Web search engines," in *Proc. WWW*, vol. 7, 2007, pp. 757–766.
[22] R. L. Cilibrasi and P. M. B. Vitanyi, "The Google similarity distance," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 370–383, Mar. 2007.
[23] Z. He, S. Gao, L. Xiao, D. Liu, H. He, and D. Barber, "Wider and deeper, cheaper and faster: Tensorized LSTMS for sequence learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.
[24] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Proc. AAAI*, 2016, pp. 2786–2792.
[25] K. S. Tai, R. Socher, and C. D. Manning. (2015). "Improved semantic representations from tree-structured long short-term memory networks." [Online]. Available: https://arxiv.org/abs/1503.00075
[26] A. Lai and J. Hockenmaier, "Illinois-lh: A denotational and distributional approach to semantics," in *Proc. 8th Int. Workshop Semantic Eval. (SemEval)*, 2014, pp. 329–334.
[27] J. Bjerva, J. Bos, R. van der Goot, and M. Nissim, "The meaning factory: Formal semantics for recognizing textual entailment and determining semantic similarity," in *Proc. 8th Int. Workshop Semantic Eval. (SemEval)*, 2014, pp. 642–646.
[28] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.
[29] S. Bird, "NLTK: The natural language toolkit," in *Proc. COLING/ACL Interact. Presentation Sessions*, 2006, pp. 69–72.
[30] M. P. Marcus and M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The penn treebank," *Comput. Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.
[31] T. Pedersen, S. Banerjee, and S. Patwardhan, "Maximizing semantic relatedness to perform word sense disambiguation," Univ. Minnesota Supercomputing Inst., Duluth, MN, USA, Res. Rep. UMSI 2005/25, 2005.
[32] L. Tan. (2014). *PYWSD: Python Implementations of Word Sense Disambiguation (WSD) Technologies [Software]*. [Online]. Available: https://github.com/alvations/pywsd
[33] Accessed: May 23, 2018. [Online]. Available: https://github.com/Manwholikespie/thesaurus-api
[34] *Wikipedia English Language Contractions*. Accessed: May 11, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Listof Englishcontractions
[35] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics, Syst. Demonstrations*, 2014, pp. 55–60.
[36] M. Honnibal, "Spacy (version 1.3.0)," Explosion AI, Berlin, Germany, Tech. Rep., 2016. [Online]. Available: https://spacy.io/
[37] J. M. Sinclair, *Looking Up: An Account of the COBUILD Project in Lexical Computing and the Development of the Collins COBUILD English Language Dictionary*. London, U.K.: Collins ELT, 1987.
[38] A. Islam and D. Inkpen, "Semantic text similarity using corpus-based word similarity and string similarity," *ACM Trans. Knowl. Discovery Data*, vol. 2, no. 2, p. 10, 2008.
[39] S. Jimenez, G. Duenas, J. Baquero, and A. Gelbukh, "UNAL-NLP: Combining soft cardinality features for semantic textual similarity, relatedness and entailment," in *Proc. 8th Int. Workshop Semantic Eval. (SemEval)*, 2014, pp. 732–742.
[40] J. Zhao, T. Zhu, and M. Lan, "Ecnu: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment," in *Proc. 8th Int. Workshop Semantic Eval. (SemEval)*, 2014, pp. 271–277.
[41] H. He, K. Gimpel, and J. Lin, "Multi-perspective sentence similarity modeling with convolutional neural networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1576–1586.
[42] M. C. Lee, J. W. Chang, and T. C. Hsieh, "A grammar-based semantic similarity algorithm for natural language sentences," *Sci. World J.*, vol. 2014, Apr. 2014, Art. no. 437162.
[43] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

ATISH PAWAR received the B.E. degree (Hons.) in computer science and engineering from the Walchand Institute of Technology, India, in 2014, and the master's degree (Hons.) in computer science from Lakehead University, in 2018. He was with Infosys Technologies, from 2014 to 2016. He is currently a Research Assistant with the DataLab, Lakehead University. His research interests include machine learning, natural language processing, and neural networks.

VIJAY MAGO received the Ph.D. degree in computer science from Panjab University, India, in 2010. In 2011, he joined the Modelling of Complex Social Systems Program, IRMACS Centre, Simon Fraser University, before moving on to stints at Fairleigh Dickinson University, The University of Memphis, and Troy University. He is currently an Assistant Professor with the Department of Computer Science, Lakehead University, ON, Canada, where he teaches and conducts research in areas including decision-making in multi-agent environments, probabilistic networks, neural networks, and fuzzy logic-based expert systems. Recently, he has diversified his research to include natural language processing, big data, and cloud computing. He has published extensively on new methodologies based on soft computing and artificial intelligent techniques to tackle complex systemic problems, such as homelessness, obesity, and crime. He has served on the program committees of many international conferences and workshops. He currently serves as an Associate Editor for *BMC Medical Informatics and Decision Making* and as a Co-Editor for the *Journal of Intelligent Systems*.