BOHR

BOHR International Journal of Computer Science
2022, Vol. 1, No. 1, pp. 1–5
DOI: 10.54646/bijcs.2022.01
www.bohrpub.com

METHODS

Measure term similarity using a semantic network approach

D. M. Kulkarni* **and Swapnaja S. Kulkarni**

Department of Computer Science Engineering, D.K.T.E. Society's Textile and Engineering Institute, Ichalkaranji, India

***Correspondence:**
D. M. Kulkarni,
kulkarni.dhanashri@gmail.com

Received: 05 May 2021; **Accepted:** 24 December 2021; **Published:** 04 January 2022

Computing semantic similarity between two words comes with a variety of approaches. This is mainly essential for applications such as text analysis and text understanding. In traditional systems, search engines are used to compute the similarity between words. In that sense, search engines are keyword-based. There is one drawback that users should know what exactly they are looking for. There are mainly two main approaches for computation, namely knowledge-based and corpus-based approaches. However, there is one drawback that these two approaches are not suitable for computing similarity between multiword expressions. This system provides an efficient and effective approach for computing term similarity using a semantic network approach. A clustering approach is used in order to improve the accuracy of the semantic similarity. This approach is more efficient than other computing algorithms. This technique can also be applied to large-scale datasets to compute term similarity.

Keywords: term similarity, multi-word expression, clustering, semantic network

Introduction

Semantic similarity measurement is the fundamental problem. Computation between two terms mainly appears in lexical semantics (1). Here, the similarity between two terms can be measured. The term in the sense of a single word or multiword expression can be taken. This technique of computing semantic similarity between words can be used in many applications, such as in case of web search or document search (2). In wed search, thousands of data are available on a very large scale. These data from the web can be used to compute term similarity. Two terms are semantically similar if they have some common attributes. For example, "apple" and "company." These two terms are semantically similar because they belong to the same category. Both terms are companies. For example, "car" and "journey." These two terms are not semantically similar but they are related. Because "journey" is an activity and "car" is a mode of transport for "journey."

WordNet (3) is a dataset consisting of thousands of words. It maintains is a relation between words. Two terms are considered semantically similar if there is an is a relation present between two terms. That is why semantic similarity is hard to model as compared to semantic relatedness. There are two main approaches to compute semantic similarity between two terms. These approaches are knowledge-based and corpus-based approaches.

In knowledge-based approach, most of the work in this space (4) depends on is a relation between words in a WordNet. is a relation between words is mandatory to compute semantic similarity between words. Corpus-based approach is a little bit different from the knowledge-based approach. In corpus-based approach, contexts of a term can be extracted from a large-scale dataset. In short, this work is mainly related to the web. Here, a corpus can be anything from a webpage or web search snippet. Here, terms are extracted from web search engines to compute semantic similarity.

There are some limitations faced by knowledge-based approach. The main problem is the limitation of taxonomy with WordNet. This approach does not cover all senses of terms. WordNet does not consist of all word sense pairs. Instead, it may contain only a single word with phrase of multiword expressions. It is impossible to compute semantic similarity between unknown terms and their senses in WordNet.

Corpus-based approach has some limitations. In this approach, semantic similarity can be computed by using search engines. A search engine uses indexing and ranking mechanisms for words. One limitation that users must know exactly what they are searching for. Otherwise, it may give ambiguous results. For example, if a user searches for an "apple," then a search engine may give all possible results for "apple," such as apple as fruit and apple as company. This may generate ambiguity. To deal with this approach, users should be clear about their concepts regarding terms and how to compute semantic similarity.

This system proposes an efficient and effective approach for computing semantic similarity between words. is a relation is present between words to compute similarity. Depending on their relation, similarity score of terms can be decided. After the completion of similarity computation, a similarity score can be generated. It generates a similarity score between 0 and 1. A system uses such a dataset which is having is a relation between two terms.

This system is more reliable and efficient to compute semantic similarity between two terms because a clustering approach is introduced. A refined approach algorithm is introduced to accurately compute semantic similarity between words. This system is also able to solve problems with ambiguous meaning.

In this study, we propose an efficient and effective framework for computing semantic similarity (a number between 0 and 1) between two terms using a large-scale, general-purpose is a network obtained from a web corpus. Below is a small sample of results:

- High similarity (synonyms): hgeneral electric and gei. Synonyms that refer to the same entity should have the highest similarity score.
- High similarity (ambiguous terms): hmicrosoft, applei, horange, and redi. Words such as "apple" and "orange" have multiple senses. However, when people compare "apple" with "microsoft," they consider "apple" in the sense of a company rather than a fruit, and when they compare "orange" and "red," they consider "orange" as a color rather than a fruit. Thus, disambiguation needs to be performed by default in similarity comparison.
- Low similarity (though share the same hypernyms in WordNet): hmusic, lunchi, hbanana, and beefi. These pairs of terms are not similar. However, in an is a network, "music" and "lunch" may both belong to concepts such as "activity," and "banana" and "beef" may both belong to concepts such as "food." We may use their distances in a handcrafted taxonomy to measure similarity, but handcrafted taxonomies have low coverage, while distances in large-scale, data-driven semantic networks are not easy to measure.

Literature survey

A new semantic relatedness measurement using WordNet features (5) by Taieb et al. system introduces a fundamental problem of computing semantic similarity between two terms. Here, information content (IC) method is used to compute similarity between words. This method also used a taxonomical feature between terms. This approach has two parts: subgraph is formed in the first part. Its descendants are counted as compartmentalization parameters. In the second part, the IC metric is integrated into a multistrategy approach.

This system that uses the IC method to evaluate semantic similarity in a taxonomy (6) by Resnik introduces an is a taxonomy to compute semantic similarity between two terms. This method is the same as the edge-counting method. The results of this system show that it produces sensible results using the IC technique.

This system that explores knowledge bases for similarity (7) by Agirre et al. introduces graph-based algorithms to compute similarity. For computing similarity, it uses WordNet along with graph-based algorithms. The WordNet-353 dataset is used to compute the similarity between words. This system is better than other traditional systems. Results show that it gives performance improvement as compared to the traditional system.

Problem statement

This study aims to generate similarity scores from given datasets, implement a basic approach, and refine an approach algorithm for computing semantic similarity between pairs of words.

Proposed work

Figure 1 illustrates the architecture of semantic similarity where a user may perform login and may give a query to the database. Here, the admin is responsible to maintain a dataset. After generating a dataset, it may upload that into the database. From a database, terms are extracted and given as input to the type-checking method. Then, the context of a term can be extracted from its type. By using a clustering algorithm according to its contexts, clusters are formed (8). Finally, by using similarity functions, the semantic similarity can be measured and output is generated.

The proposed system is designed and developed with following modules.

Module 1: Candidate set of words from data dictionary

A dataset may consist of a collection of words. It consists of more than 1,00,000 words, where words may have multiple definitions. It may contain phrases P, which is a word or

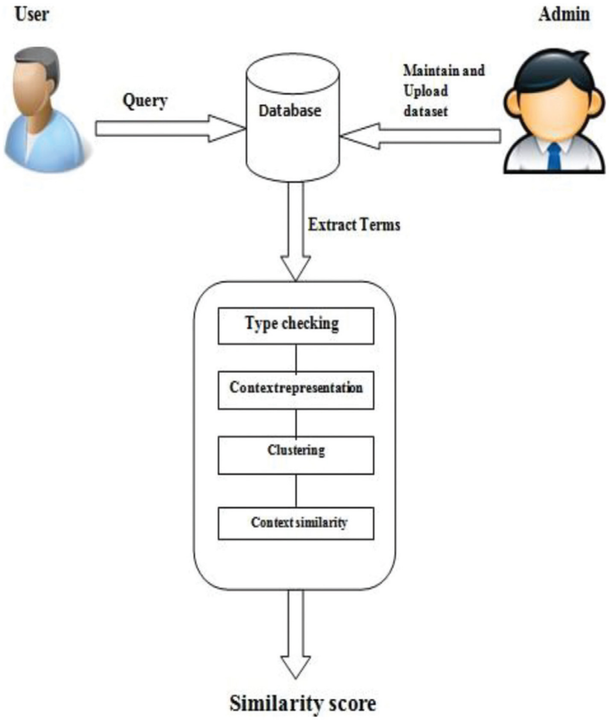


FIGURE 1 | Architecture of semantic similarity.

sequence of words. Words in a dataset can be related to each other by their type, such as a dataset may consist of synonyms set, antonyms set, hypernym set, or hyponym set of words. For maintaining a dataset, an algorithm is used which takes a sequence of terms as an input. The output of this method is a set of words. If a required dataset is available, then one semantic similarity can be measured. Otherwise, dataset can be generated by using algorithms.

Module 2: Type checking

The first step while computing a semantic similarity is to check the type of a given term. The type of a term can be either an entity or a concept. For type checking of a term, two things are required: an entity or concept set and an is a relation between terms. If an is a relation is maintained between terms, then the hypernym term is a concept term. The hyponym term is an entity term. If no is a relation is maintained between terms, then its type can be decided individually. For example, the concept of the terms "Apple and Microsoft" is company.

Module 3: Context representation

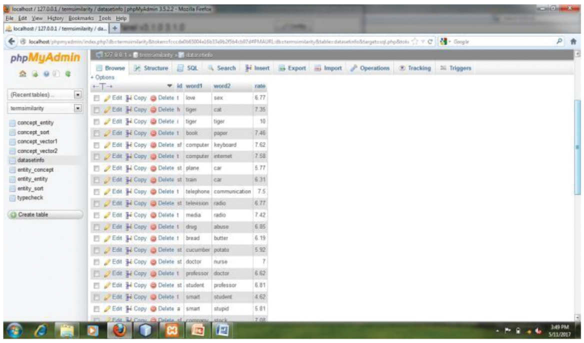
A context of a given term can be extracted from its type and is dependent on its type so that a type of a term can be input into it. A context can be an entity if a term is a concept. If a given term is an entity, then its context can be a concept. For example, the concept contexts of the term "Apple" are fruit, company, food, seasonal fruit, and tree.

Module 4: Concept clustering

A concept clustering algorithm is added into a refined approach algorithm as a part of it. For finding similarity, a clustering algorithm was implemented as a part of the refined approach algorithm. To identify multiple senses of terms, the K-medoid clustering algorithm is used. A clustering algorithm takes a collection of concepts as an input. By using this clustering algorithm, similar contexts or senses of a term are grouped together. For example, fruit, seasonal fruit, and tree fruit are grouped together into one cluster because all contexts of the term "Apple" have the same sense.

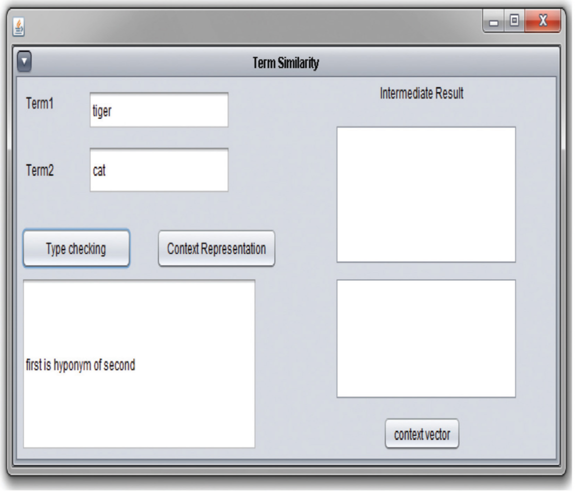
Module 5: Context similarity

To estimate the similarity between two contexts, a similarity function F(.) can be used. The similarity can be measured as Sim(Tt1, Tt2) = F(Tt1, Tt2). The similarity function F(.) can be any one of the evaluation functions, such



id	word1	word2	rate
t	love	sex	6.77
h	tiger	cat	7.35
i	tiger	tiger	10
t	book	paper	7.46
sf	computer	keyboard	7.62
t	computer	internet	7.58
st	plane	car	5.77
st	train	car	6.31
t	telephone	communication	7.5
st	television	radio	6.77
t	media	radio	7.42
t	drug	abuse	6.85
t	bread	butter	6.19
st	cucumber	potato	5.92
st	doctor	nurse	7
t	professor	doctor	6.62
st	student	professor	6.81
t	smart	student	4.62
a	smart	stupid	5.81
sf	company	stock	7.08

FIGURE 2 | Candidate set of words.



Term1 tiger

Term2 cat

first is hyponym of second

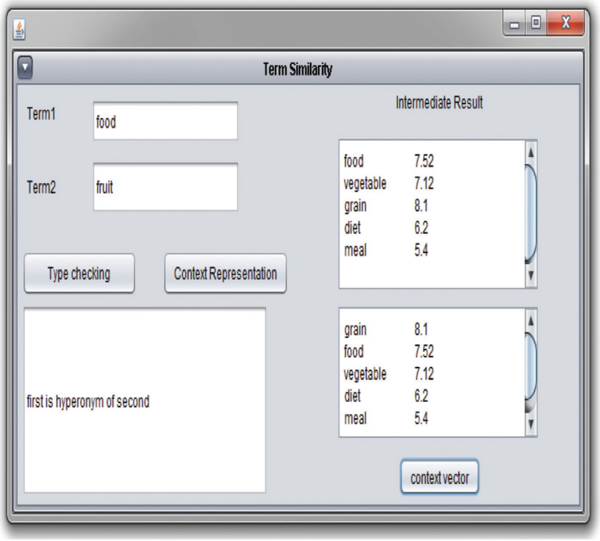
FIGURE 3 | Type checking.

as cosine and Jaccard. Other methods used for finding similarity are Max: sim(Tt1; Tt2), Average: sim(Tt1; Tt2), and Weighted: sim(Tt1; Tt2).

Experimental result

Module 1

Figure 2 shows the dataset that consists of the collection of word pairs. Each word pair has been assigned an id and a type. The WordSim-353 dataset is used.



Term1: food

Term2: fruit

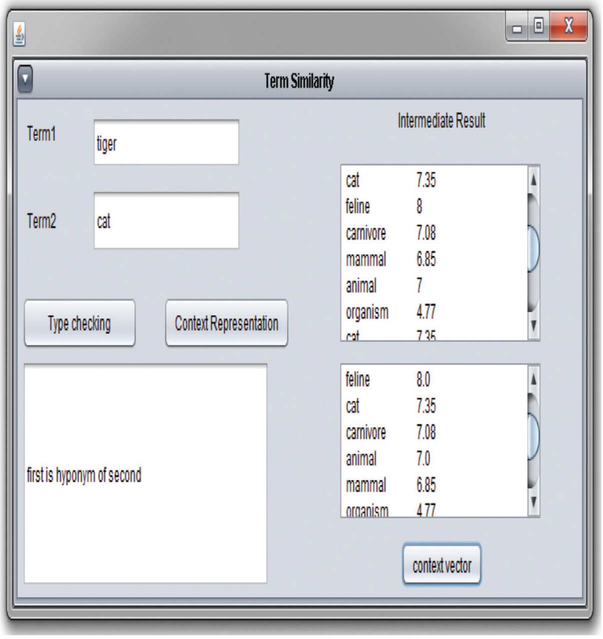
Intermediate Result

```
food       7.52
vegetable  7.12
grain      8.1
diet       6.2
meal       5.4
```

```
grain      8.1
food       7.52
vegetable  7.12
diet       6.2
meal       5.4
```

first is hyperonym of second

FIGURE 4 | Context representation (id = hr).



Term1: tiger

Term2: cat

Intermediate Result

```
cat        7.35
feline     8
carnivore  7.08
mammal     6.85
animal     7
organism   4.77
cat        7.35
```

```
feline     8.0
cat        7.35
carnivore  7.08
animal     7.0
mammal     6.85
organism   4.77
```

first is hyponym of second

FIGURE 5 | Context representation (id = h).

Module 2: Type checking

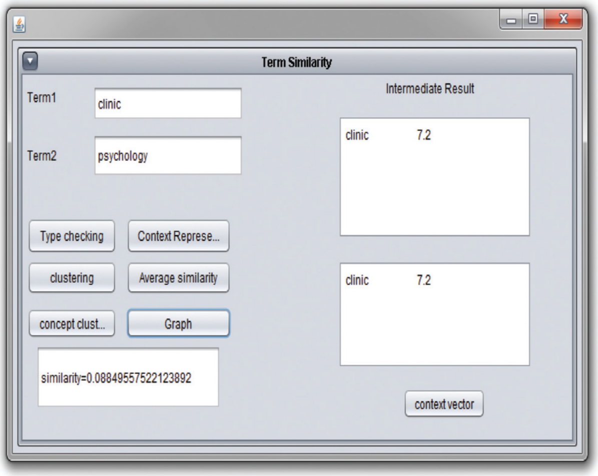
Figure 3 shows the result of type checking in which two-word pairs are given as input, having an is a relation between words.

Module 3: Context representation

Figures 4, 5 show the result of context representation in which two terms are given as input. The context of word pair can be determined according to its id assigned in a dataset.

Module 4: Concept clustering

Figure 6 shows the result of clustering in which two terms are given as an input. Clusters of word pairs can be generated according to their id and the family from which they belong.



Term1: clinic

Term2: psychology

Intermediate Result

```
clinic     7.2
```

```
clinic     7.2
```

similarity=0.08849557522123892

FIGURE 6 | Concept clustering.

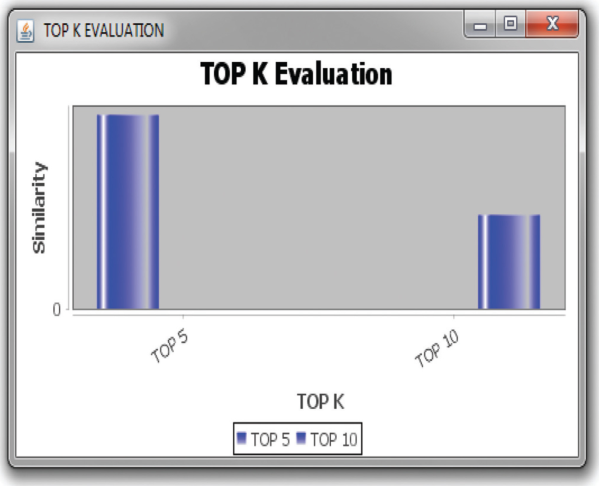


FIGURE 7 | Similarity of words.

Module 5: Context similarity

Figure 7 shows the result of similarity between two words in terms of graphical representation.

Conclusion

This is an efficient and effective approach for computing semantic similarity between terms. is a semantic network is present between pairs of words. A concept clustering algorithm was introduced to avoid ambiguous terms. Finally, the maximum similarity function is used to compute the similarity between two terms. This method is efficient enough to apply on large-scale datasets. Future work on this system should focus on is how to apply the same technique to short text categorization.

References

1. Budanitsky A, Hirst G. Evaluating wordnet-based measures of lexical semantic relatedness. *Comput Linguist.* (2006) 32:13–47.
2. Geartner M, Rauber A, Berger H. Bridging structured and unstructured data via hybrid semantic search and interactive ontology-enhanced query formulation. *Knowl Inf Syst.* (2014) 41:761–92.
3. Miller GA. WordNet: a lexical database for English. *Commun ACM.* (1995) 38:39–41.
4. Resnik P. Using information content to evaluate semantic similarity in a taxonomy. *Proceedings of the 14th International Joint Conference for Artificial Intelligence (IJCAI-95).* (1995). p. 448–53.
5. Taieb MAH, Aouicha MB, Hamadou AB. A new semantic relatedness measurement using wordnet features. *Knowl Inform Syst.* (2014) 41:467–97.
6. Hearst MA. Automatic acquisition of hyponyms from large text corpora. *Proceedings of the 14th Conference Computational Linguistics*. Nantes: (1992). p. 539–45.
7. Song Y, Wang H, Wang Z, Li H, Chen W. Short text conceptualization using a probabilistic knowledgebase. *Proceedings of the 22nd International Joint Conference artificial intelligence*. Washington, DC: (2011). p. 2330–6.
8. Wang W, Tang B, Zhu C, Liu B, Li A, Ding Z. Clustering using a similarity measure approach based on semantic analysis of adversary behaviors. *Proceedings of the 2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC)*. Hong Kong: (2020).
9. Wang Y, Li H, Wang H, Zhu KQ. Concept-based websearch. *Proceedings of the 31st International Conference Conceptual Model. ER*. (2012). p. 449–62.
10. Rada R, Mili H, Bichnell E, Blettner M. "Development and application of a metric on semanticnets,". *IEEE Trans Syst Man Cybern.* (1989) 9:17–30.
11. Wu W, Li H, Wang H, Zhu KQ. Probase: a probabilistic taxonomy for text understanding. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. (2012). p. 481–92.
12. Shaw R, Datta A, VanderMeer D, Dutta K. Building a scalable database- driven reverse dictionary. *IEEE Trans Knowl Data Eng.* (2013) 25:528–40.
13. Agirre E, Cuadros M, Rigau G, Soroa A. Exploring knowledge bases for similarity. *Proceedings of the 7th International Conference on Language Resources Evaluation (LREC'10).* Valletta: (2010). p. 373–7.