A Visual Semantic Relations Detecting Method Based on WordNet

Wenxin Li[1], Tiexin Wang[1,2(✉)], Jingwen Cao[1], and Chuanqi Tao[1,2,3]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, 29#, Jiangjun Road, Jiangning District, 211106 Nanjing, China
{freedomtot, tiexin.wang}@nuaa.edu.cn, caojingwen1028@126.com, t-chuanqi@163.com
[2] Ministry Key Laboratory for Safety-Critical Software Development and Verification, Nanjing University of Aeronautics and Astronautics, Nanjing 211100, China
[3] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, People's Republic of China

Abstract. In order to implement automatic inference, this paper proposes a visual semantic-relations detecting method (VSRDM) based on WordNet. WordNet is an excellent relational dictionary, but it lacks deep semantic topology function because of its index-based text storage structure. As a graphical database, Neo4J provides visualization of its internal data. Since the abstract data structure in WordNet matches Neo4J's ternary storage structure, it is very suitable to map WordNet completely with Neo4J graph instance. This paper studies how to fully describe WordNet in Neo4J through a ternary structure. Neo4J stores the data as graphs (nodes and edges) and provides certain native graph algorithms to search the data. The speed of matching query between nodes is varying linearly with the number of nodes, so the efficiency of basic operation is guaranteed. With the help of Neo4J, VSRDM works as a semantic dictionary providing relationships matching, reasoning auxiliary and other functions.

Keywords: WordNet · Neo4J · Semantic relations · Knowledge-Driven

1 Introduction

In a server, when the CPU has a high load, skilled maintenance staff will initially locate the problem according to experience, and gradually deduce and troubleshoot. The staff conducts daily maintenance and troubleshooting based on each log that reports an error. How to use tools to automatically interpret or infer potential situations based on current states or conditions? Admittedly, using deep learning to train inferential models based on massive data is an effective mode, but how to make a reasonable judgment with limited data sets? WordNet is a tool that can help complete the above work. However, since WordNet is only a collection of structured data, a powerful processing engine is required to dig its potential.

X. B. Zhai et al. (Eds.): MLICOM 2019, LNICST 294, pp. 465–476, 2019.
https://doi.org/10.1007/978-3-030-32388-2_40

WordNet is an English dictionary based on cognitive linguistics designed by experts from Princeton University [1]. It has rich semantic relations, so it can be used as ontology and plays an important role in related fields.

However, most of the current applications built up on WordNet are purely semantic computing, which cannot give full play to the ability of WordNet as ontology. The native WordNet only provides functional interface and service as a dictionary. We urgently need a method to describe WordNet in a structured way and give full play to it as ontology.

Currently, Neo4J has been employed quite often in industry applications. For example, it has played a significant role in financial domain application [2], user map in social field and enterprise relationship map. The combination usage of Neo4J and WordNet will play a certain role as an easy-to-use ontology in an increasingly intelligent world.

In order to make full use of WordNet, this paper proposes a visual semantic relations detecting method based on WordNet. The mapping rules from WordNet to Neo4J are defined by analyzing the structure of the two. In addition, several simple application methods of WordNet powered by Neo4J are put forward to play a role of introducing jade. Figure 1 shows the overall structure of VSRDM.

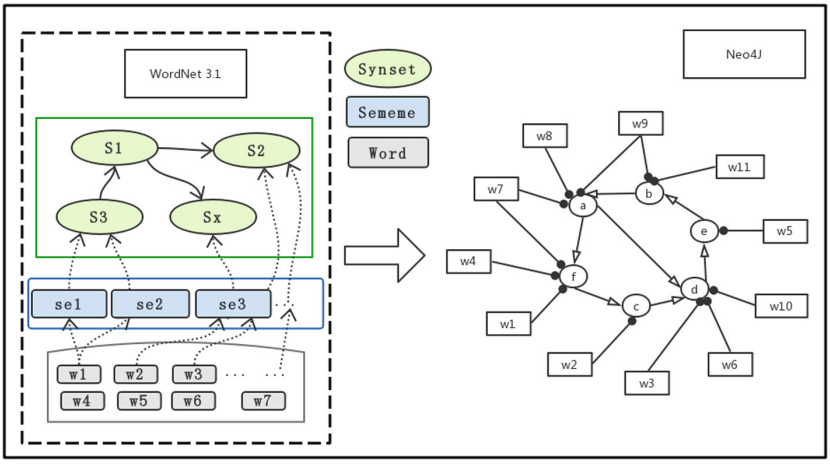

Fig. 1. The overall structure of VSRDM

The structure of this paper is as follows. The second section presents the technology foundation (i.e., WordNet and Neo4J). The third section analyzes the data structure of WordNet and shows the structure of the Neo4J database instance. VSRDM is implemented based on the analysis of these data structures. The fourth section shows a use case of VSRDM. Finally, the fifth section draws a conclusion.

2 Related Work

As a cognitive linguistics (English) dictionary, WordNet overcomes the problem of information organization in traditional dictionaries, it is suitable for semantic computing. Its core lies in extracting the semantic relationship between words, realizing the visualization through the concept of synonym set, and forming a semantic relationship network [3]. Each synonym set represents a basic semantic concept in which words are linked to each other in an indirect way. In WordNet, there are noun semantic network, verb semantic network, adjective semantic network and adverb semantic network [1].

At present, researches on WordNet are mainly concerning on semantic computation. In view of the present word similarity algorithm is widespread in the single source of information, the results of nonlinear on the high side, and the computing performance and efficiency of the inconsistent defect [4]. Based on the upper and lower relation diagram of WordNet synonym set, an algorithm is implemented which can calculate the similarity between English - English, Chinese - English and Chinese - Chinese words.

The affective dictionary constructed by HowNet is of high accuracy and availability in the construction of catering review affective dictionary. This proves that similar research can be done in WordNet [5].

WordNet is often used in the analysis of concatenation of multiple languages. In order to improve the adaptability of WordNet in different language environments, researchers in many countries carried out researches on how to add additional relevant clauses to increase the efficiency and accuracy of queries [6] (Table 1).

Table 1. Comparison of VSRDM and related studies.

	Semantic detection	Knowledge reasoning	Reasoning auxiliary	Knowledge graph construction
Research [7]	Y	–	–	–
Research [8]	–	Y	Y	–
Research [9]	Y	–	–	–
VSRDM	Y	–	Y	Y

Neo4J is a graph database specially used for network graph storage. It has higher mass data processing speed, more intuitive data, more flexible data storage and stable computing efficiency [10]. When the data volume and data association reach a certain degree, the traditional relational database becomes gradually weak, and Neo4J can deal with it stably.

Compared with the traditional SQL statement, the Cyper language used by Neo4J is more intuitive to express relationships. New data is stored as edges, nodes, edge attributes and node attributes without considering the structure of tables. Neo4J operates at a consistent speed thanks to its underlying graph storage structure and optimization algorithm based on graph data structure.

Many investigators study the embedded application of graph database by taking the graph of author collaboration in the field of education as an example to provide a fast and dynamic storage method. Research shows that graphic data is easy to use in database creation, query, update and other aspects, especially suitable for processing a large number of complex, dynamic, interconnection, low structured data, is an effective method to solve the social network, information visualization and other fields of mass related data storage [11].

3 An Overview of VSRDM

3.1 Preparation Work

WordNet Data Structure. WordNet enables to classify words by explanation, and a group of words with the same meaning is named as synset. WordNet provides a brief definition for each synset and records the semantic relationships between different synsets. There are four synonymic networks, namely, noun network, verb network, adjective network and adverb network.

The underlying data structure of WordNet consists of entry, sememe, and synonym sets, each of which has its own entry ID and is stored in specific text file. Each entry contains its dependent sememe ID, each sememe points to its dependent synonym set file, synonym set ID, and each synonym set entry contains its definition, usage, and relationship to other synonyms (Table 2).

Table 2. Synonym set relationships in WordNet.

Class	Core relationship
Noun	*Hyponymy/Merony/ComponentOf/Holonymy/Antonymy/Attribute/...*
Verb.	*Antonymy/Entailment/Cause/AlsoSee*
Adj.	*Similarity/Relational/AlsoSee/Attribute*
Adv.	*Antonymy/DerivedFrom*

Diagram Storage Structure of Neo4J. Neo4J stores user-defined nodes and relationships in a graph way, which can start from a certain nodes and find out the relationships between the initial and target nodes by taking advantage of the inter-node relationships with high efficiency.

The underlying storage way of Neo4J determines the capable direction. Figure 2 shows the case of supply chain management provided by Neo4J official website. Due to the large quantities and complicated types of factors involved in the supply chain, the analysis and decision-making of the supply chain is quite complicated. Therefore, it is quite significant to organize a clear overview and understanding of the whole supply chain. Neo4J enables to do us a favor to acquire a great understanding for supply chain management.

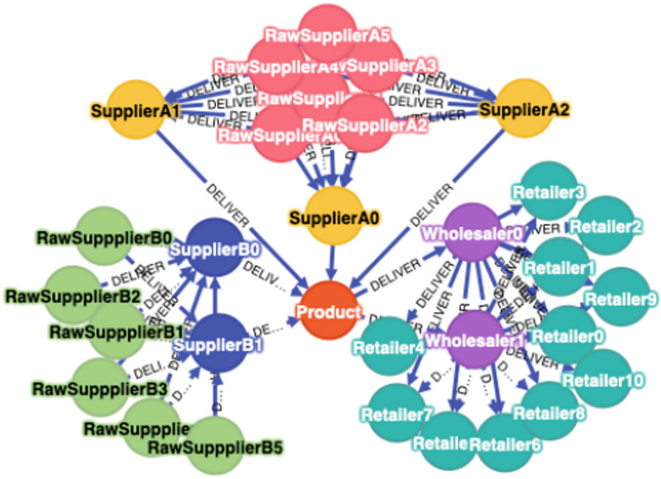

Fig. 2. Supply chain management [12].

3.2 Mapping Rule Definition

The mappings between WordNet and Neo4J can be realized based on the above analysis. The words and synonym sets are nodes, but they have different attributes of key-value pairs, such as lemma, WID (word's ID), sense attributes, the use of case examples, SID. Word related with synonym set, synonym set associated with synonym set ought to have connections, and these two kinds of connections also enable to possess various attributes, which respectively corresponds to the word and a synonym set.

While building the mappings, the difficulty exists in how to achieve a complete semantic mapping of heterogeneous data structure. This method applies the idea of three-segment mapping to achieve the semantic preservation of the mapping of WordNet dictionary to Neo4J database in a nearly complete approach.

The first step is matching synonym sets, in which all kinds of relational mappings between synonym aggregation points are not generated at the same time.

The second step is matching synonym set relational mappings. In this mapping process, breadth-first algorithm is used to traverse each synonym set associated with a specific relation based on the association set of each synonym set, and the connections between synonym aggregation points in the corresponding Neo4J is generated, and the related connection properties are filled in.

The third step is matching word mappings, in which all the words in WordNet are traversed and each word's sememe set is traversed, and the words are bound to the corresponding synonym set in fixed attribute connection based on the synonym set specified in the sememe (Table 3).

Table 3. Details of the three-stage matching.

	Processing content	No. new node	No. new connection
1st	Thesaurus network node	175979	0
2nd	Thesaurus network relationships	0	207016
3rd	The children of the synonym word set network	155327	0

3.3 Transaction from WordNet to Neo4J Instances

Due to the nature of Neo4J, the node mapping for WordNet must be implemented in the order of three-segment mapping.

Synonym Set Mapping Stage. In the first stage, the mapping of synonym sets is mainly to ensure that the second-stage mapping of the relationship between synonym sets and the establishment of the synonym aggregation point relationship will not generate a one-sided relationship, and the third-stage mapping of word nodes will not indicate null nodes. In this study, a WordNet parsing engine is maintained to generate Cyper statements for constructing synonym nodes. The general process is described by the following pseudo-code:

```
FOREACH(synset:ITERATOR)
CreateState+=createCyperOfSynset(synset)
Val STATE = CreateState
Neo4J_DRIVER.run(STATE)
```

In the first step, a fixed Synset type Synset is generated for each synonym assembly point, the name being determined by the Synset's corresponding lexical, and each synonym assembly point has the synonym set ID attribute, SID, and the description attribute Gloss. The generated Cyper statement is roughly as follows.

```
CREATE (a:Synset:[lexical1]{SID:[offset1],Gloss:[gloss1]})
+ CREATE (b:Synset:[lexical2]{SID:[offset2],Gloss:[gloss2]}) + ...
```

Synonym Set Semantic Relation Network Mapping Stage. In the second phase, the core of WordNet is implemented according to BFS: the mapping of semantic relation network between synonyms. During the construction of the Cyper statement by the interpretation engine, the MATCH and WHERE sub-statements are de-reprocessed, and the general process of generating the Cyper statement was described by the following pseudo-code.

FOREACH(subSynset:ITERATOR)

CreateState+=createCyperOfSubSynset(subSynset)

MatchState+=matchCyperOfSubSynset(subSynset)

WhereState+=whereCyperOfSubSynset(subSynset)

Val STATE = MatchState + WhereState + CreateState

Neo4J_DRIVER.run(STATE)

The second phase starts with each synonym marshalling point and traverses all the synonym marshalling points which have a direct semantic relationship with them. Each corresponds to a MATCH clause, a WHERE clause, and a CREATE clause. The connection between each generated synonym assembly point has various connection types according to the semantic relation, such as hyponymy, hypernymy, meronymy, component_of and so on. The generated Cyper statement is roughly as follows.

MATCH (s1:[lexcal_file_name]) , (s[offset1]:[lexxical_file_name1]) , ...

+ WHERE (s1.SID=[offset]) AND (s[offset1].SID=[offset1]) AND...

+ CREATE (s1)-[:[pointer1]]->(s[offset1]) + ...

Mapping Stage of Lexeme and Word Node. In the third stage, all words are traversed to generate corresponding word nodes, and word to synonym set binding is realized based on sememe. The explain engine will build MATCH and WHERE statements based on the SID of the synonym set pointed to in the sememe and generate CREATE statements that build the connections between the word and the synonym set. The general process of generating Cyper statements is described by the following pseudo-code.

FOREACH(synset:SENSE_LIST)

MatchState+=matchCyperOfSynset(synset)

WhereState+=whereCyperOfSynset(synset)

CreateState+=createCyperOfWordRelation(word,synset)

Val STATE = MatchState + WhereState + CreateState

Neo4J_DRIVER.run(STATE)

In the stage of building the word and synonym assembly point relationship, the creations of the word node and the related synonym set are resolved after the traversal of the sememe of the current word, where the connection type of the word to the synonym set is the fixed type word2synset. The Cyper statements built during the mapping process are roughly as follows.

```
MATCH (s[offset1]:Synset:[lexical1]) , (s[offset2]:Synset:[lexical2]) ...
+ WHERE (s[offset1].SID=[offset1]) AND (s[offset2].SID=[offset2]) ...
+ CREATE(word:Word{lamma:[lemma]})
+ CREATE (word)-[:word2synset]->(s[offset1]) + ...
```

3.4 Performance Optimization

Theoretically, the first stage and the third stage shall take the same time to create nodes. However, the reality is that, in the third stage, the word node creation takes two times higher than the first stage of the time-consuming. The reason is that Cyper statement need to find them (MATCH) if expect to link the two existing nodes.

While creating semantic relationships between a single word and multiple synonyms, the mapping node of each synonym set needs querying first. If multiple words are semantically related to the same synonym set, there will be too many synonyms in the current implementation to repeat the query process, which is also the reason for the huge time consumption gap between the third stage and the first stage. Similar problems were encountered in the second phase of creating semantic relationships for the same reason.

One approach to solving the problem of repeated query of nodes is to take advantages of DFS algorithm to create multiple relationships between multiple nodes in the same Cyper statement in a complete DFS recursive process, so as to reduce repeated query of the same node.

Another solution is that although the relationships between the three stages must be synchronous, any creation operation in each stage is thread-safe, that is, the creation of nodes and connections within each stage can be realized by multi-thread creation, thus improving the operation efficiency (Table 4).

Table 4. A comparison of matching processes.

Stage	Before optimization	After optimization
1st	1260 s	900 s
2nd	1800 s	1320 s
3rd	1400 s	1100 s

4 Functional Design

4.1 Maintain Native Dictionary Functionality

WordNet officially provides a web service, which supplies a complete dictionary function based on WordNet data and user inputs. In the mapped Neo4J entities, this capability is still maintained. Figure 3 is the web dictionary service provided by the official WordNet website.

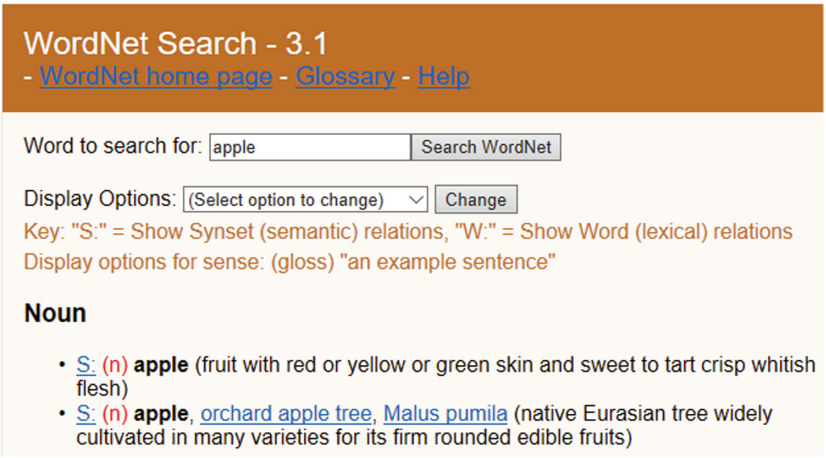

Fig. 3. Dictionary function [13]

4.2 Dictionary Functional Verification

Only use a single layer of semantic detection to query the nodes directly related to "apple" and compare it with the dictionary function provided by WordNet website. The Cyper statement is:

*MATCH (n:Word)-[*1..2]-(k) WHERE n.lamma="apple" RETURN n,k*

Based on this statement, it successfully found two synonym aggregation points directly connected with "apple", and the results can be displayed by JSON and visualization. The following Fig. 4 respectively shows the matching results of simulated dictionary query.

Fig. 4. Dictionary functionality based on Neo4J - "apple"

As shown in the figure, it successfully found two sets of synonyms of apple's direct semantic relationship by mapping the Neo4J instance, namely the junction of the noun_food type and the noun_planet type, and after comparing the JSON data and the result data of the webpage query, all the related information will be fully expressed.

4.3 Relational Query

In term of Neo4J, multi-level words and nested relationships between words can be visually verified and matched.

Look for word pairs for all semantic relationships.

*MATCH item=(word1 { lamma :?...})-[*n...{...}]-(word2{ lamma :?...}) RETURN item*

The meaning of this statement is to find the relationships between all the words i.e., *word1* and *word2*. The number of iterations of the relationship is specified by '*n*', which can be limited or determined.

Verify multilevel relationships of word pairs.

MATCH (word1)-[]->(synset1)-[]-...-[]<-(synsetn)-(word2)

The process of semantic detection is the analysis of the relationship between pairs of words. The analysis of the relationship between pairs of words generally sets the specified default value and has no other qualification conditions.

In order to implement a semi - automated generation of template - based Cyper statements, the implementation of relational - based semantic detection requires supplying a common interface to further extend the function of the method.

4.4 Relational Query Function Validation

To facilitate the description, it assumes that the following application example, using the Cyper statement:

*MATCH (n:Word)-[*1..4]-(k:Word) WHERE n.lamma="spring" AND k.lamma="winter" RETURN n,k*

In common sense, *winter* and *spring* are two kinds of seasons. The difference between them is that everything grows in spring while winter is cold and silent. Figure 5 shows the query result of the Cyper statement.

The blue node is the synonym gathering point, and each synonym set describes, from left to right, the "*season of recovery*", *"one of the natural periods of the year divided by atmospheric conditions, the winter solstice, and the vernal equinox",* and *"the coldest season"*. The relation of the three synonyms is apparently that the two nodes are the hyphen of the middle node. Moreover, by comparing the description of the synonym set with the familiar facts, the correctness of the relationship between the word pair matching is illustrated.

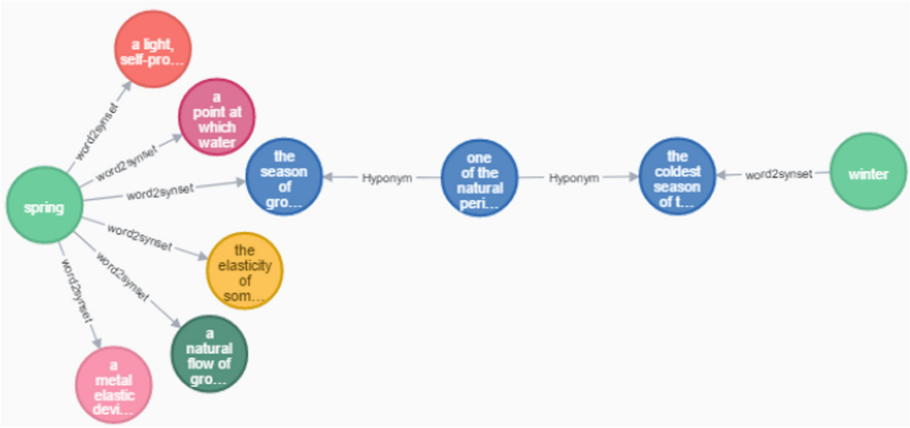

Fig. 5. Semantic relationship detection - "spring" and "winter". (Color figure online)

4.5 Reasoning Auxiliary

Regarding WordNet as ontology (semantic relation as the core and synonym set as the foundation), it has a clear description of semantic relation causality, hyponemical relation, inclusion relation, equivalence relation and so on. But also because of the natural structure of WordNet, the derivation of the complex and deep iterative semantic relations can only be detected by the traversal and judgment of relations. Fortunately, WordNet powered by Neo4J, provides the ability to truly network knowledge. The optimized graph algorithm in Neo4J can quickly and even visually show all the internal relations between things, and further realize the auxiliary functions of semantic reasoning.

5 Conclusion

This paper compares and analyzes the differences and connections between WordNet data structure and Neo4J graph instance storage structure, and describes a complete mapping scheme from WordNet to Neo4J graph instance. VSRDM is implemented and a use case of it is shown. Users can easily use VSRDM to achieve semantic relationships detection, semantic-based reasoning and knowledge graph construction.

However, the core of VSRDM, WordNet, has not been updated iteratively for a long time, so VSRDM cannot provide deep semantic analysis for vertical domains. On the other hand, the reasoning auxiliary function of VSRDM lacks automatic relation link extraction, and it is easy to have redundant relations in the process of relation inference. These problems require further study.

In the future, it is planned to expand the ontology adopted by VSRDM into domain-cross ontologies to further improve the ability of semantic relations detection of VSRDM.

Acknowledgement. This work was supported by the "Fundamental Research Funds for the Central Universities Nos. 3082018NS2018057", the National Natural Science Foundation of China (61872182), the National Natural Science Foundation of China under Grant No. 61402229 and No. 61602267, the Open Fund of the State Key Laboratory for Novel Software Technology (KFKT2018B19) and the Open Fund of the Ministry Key Laboratory for Safety-Critical Software Development and Verification (1015-XCA1816401).

References

1. Miller, G.A.: WordNet: a lexical database for English. Commun. Assoc. Comput. Mach. **38**(11), 39–41 (1995)
2. Fellbaum, C., Miller, G.: WordNet: an Electronic Lexical Database. In: Lal, M. (ed.) Neo4J Graph Data Modeling (2015). 1998
3. Fellbaum, C., Miller, G.: WordNet: An Electronic Lexical Database. MIT press, Cambridge (1998)
4. Kashyap, L., Joshi, S.R., Bhattacharyya, P.: Insights on Hindi WordNet coming from the IndoWordNet. In: Dash, N.S., Bhattacharyya, P., Pawar, J.D. (eds.) The WordNet in Indian Languages, pp. 19–44. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-1909-8_2
5. Rong, X.: word2vec parameter learning explained. In: Computer Science (2014)
6. Belalem, G., et al.: Arabic query expansion using WordNet and association rules. Int. J. Intell. Inf. Technol. 51–64 (2017)
7. Duong, T.H., Tran, M.Q., Nguyen, T.P.: Collaborative Vietnamese WordNet building using consensus quality. Vietnam J. Compu. Sci. **4**(2), 85–96 (2017). https://doi.org/10.1007/s40595-016-0077-x
8. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: International Conference on World Wide Web (2007)
9. Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet: similarity - measuring the relatedness of concepts. In: National Conference on Artificial Intelligence AAAI Press (2004)
10. Drakopoulos, G., Gourgaris, P., Kanavos, A.: Graph communities in Neo4J. Evolving Syst. **6**, 1–11 (2018)
11. Lu, H., Hong, Z., Shi, M.: Analysis of film data based on Neo4J. In: IEEE/ACIS International Conference on Computer & Information Science (2017)
12. Neo4J Homepage. https://Neo4J.com. Accessed 05 Mar 2019
13. WordNet Homepage. https://WordNet.princeton.edu. Accessed 06 Mar 2019