J. Parallel Distrib. Comput. 118 (2018) 22–33

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Scalable Distributed Semantic Network for knowledge management in cyber physical system

Shengli Song[a,*], Yishuai Lin[a], Bin Guo[b], Qiang Di[c], Rong Lv[c]

[a] Software Engineering Institute, Xidian University, Xi'an, China
[b] School of Computer Science, Northwestern Polytechnical University, Xi'an, China
[c] School of Computer Science and Technology, Xidian University, Xi'an, China

HIGHLIGHTS

- We design multiple order semantic parsing to analyze heterogeneous data.
- We propose a hierarchical scalable Distributed Semantic Network to express knowledge.
- We propose a MapReduce-based PDC framework for knowledge base construction.
- Distributed Semantic Network is very suitable for PDC framework in CPS.
- We get much more knowledge and a better knowledge base in less time.

ARTICLE INFO

Article history:
Received 31 December 2016
Received in revised form 20 October 2017
Accepted 17 November 2017
Available online 27 November 2017

Keywords:
Parallel and distributed computing
Knowledge management
Distributed semantic network
MapReduce framework
Cyber physical system

ABSTRACT

The remarkable growth of emerging technologies and computing paradigms in cyberspace and the cyber physical systems generate a huge mass of data sources. These different autonomous and heterogeneous data sources can contain complementary or semantically equivalent information stored under different formats that vary from structured, semi structured, to unstructured. These heterogeneities influence on data semantics and meaning. Therefore, knowledge management became more and more difficult and sometimes fruitless. In this paper, we propose a new scalable model, named Distributed Semantic Network (DSN), for heterogeneous data representation and can extract more semantic information from different data sources. We use the prior knowledge of WordNet and Wikipedia to scale out DSN horizontally and vertically. Furthermore, we proposed a MapReduce based framework to construct the knowledge base more effectively in Parallel and Distributed Computing (PDC). The experimental results show that DSN can better model the semantic information in the text. It can extract a larger amount of information from the text with a higher precision, achieving 34% increase in quantity and 15% promotion on precision than the best-performing alternative method on same datasets. On the three datasets, our proposed PDC framework shorten the process time by 5.8–11.5 times.

1. Introduction

Cyber Physical System (CPS) can be viewed as a large networked system, distributed geographically with each CPS site having hybrid computing patterns. Because of the collective might of CPS nodes, the capabilities of CPS could have far reaching effects in finding solutions to grand challenges in fields that combine science, engineering, and economics. Thus, CPS applications will require resources from different domains. In the absence of central authority to moderate and manage resource requests for CPS services, there is a need to facilitate the coordination of services and cooperation to share resource knowledge among resource providing nodes in CPS. There is a large variety of heterogeneous data stored in CPS nodes, includes traditional relation database that from different software vendors, including different types of NoSQL database, and different types of file, all of the data resources grow rapidly, become a great repository of data. For meeting the requirements of knowledge services, we need to extract the knowledge implied in heterogeneous data, represent and integrate the knowledge to support knowledge system.

Knowledge Base Construction (KBC), the process of populating a knowledge base with semantic information from heterogeneous data, is an important and difficult task. In recent years, several large-scale knowledge bases have been constructed, such as YAGO [31], NELL [7], DBpedia [3], IBM's Watson [13] and Microsoft's EntityCube [39]. However, the rapid growth of huge heterogeneous textual data, stored in different data sources with

[*] Corresponding author.
E-mail address: shlsong@xidian.edu.cn (S. Song).

https://doi.org/10.1016/j.jpdc.2017.11.014
0743-7315/© 2017 Elsevier Inc. All rights reserved.

different format and structure (e.g. mobile computing, social computing, Internet of Things, etc.), make it urgent to find a way to acquire and management knowledge from these data quickly. It is clear that building a distributed parallel framework to extract knowledge and construct knowledge base meet this requirement efficiently. Simultaneously, the present expression of heterogeneous textual data is weak semantic for random distribution so that it is not conducive to Parallel and Distributed Computing (PDC). It is obvious that there is a demand for a good data representation method with strong semantic can help parallel extraction of knowledge for better performance.

At present, there are two main methods for data representation: vector-based representation and graph-based representation. Vector-based representation uses word frequency or context information to express the data in digital form. Graph-based representation uses nodes to represent terms and uses edges to represent the specific relationship between the nodes. Although the vector-based and graph-based representation methods can express the existing features of heterogeneous data to some degree, they have their respective shortcomings. For word vector, it can convert the text into a vector according to the similarity between the text words, but cannot reflect the order relation of the words in the text data and capture the role that the words play in the text. For graph model, it can express the order or structural feature of the statement on a single level, but cannot reflect the similarity of the text vocabulary and the semantic relations between the words, such as the relation of generalization and specialization.

In this paper, we propose a hierarchical graph based Distributed Semantic Network (DSN) representation model for knowledge extraction and knowledge base construction with MapReduce framework in PDC. Distributed semantic representation for data is a critical component of many networks based knowledge based systems. It can be used to describe the semantic distribution in textual data. The new DSN overcomes the sparsity of natural languages by representing words with high-dimensional vectors in a continuous space and represents the meaning of data by distributing the conceptual vectors to different layers according to their information entropy. It not only captures the word order relation and subject–predicate–object structure but also adds the relationship between the hypernym and hyponym in the text. Also, we use the prior knowledge of WordNet and Wikipedia to scale out DSN with synonym nodes and nearest hypernym nodes, which is equivalent to capturing the similarity relation and inclusion relation between text terms. In a word, our proposed model can express complete and richer semantic information, which can reflect the distributed features for parallel KBC in CPS.

The scalable architecture of DSN is pretty suitable for information extraction in a distributed parallel way. To achieve this goal, we propose a MapReduce-based distributed parallel framework to extract as many as possible triples in the form of <subject, predicate, object> from expanded DSN and use triples for KBC.

The main contributions of this paper can be summarized as follows:

(1) We design Multiple Order Semantic Parsing (MOSP) to analyze heterogeneous data, that is, each piece of semantic information implied in the text is represented by a set of nodes and edges associated.

(2) We propose a hierarchical graph based scalable DSN to express semantic information implied in the text and absorb more prior external knowledge for expansion.

(3) We propose a MapReduce-based framework to extract semantic information from DSN and manage it for knowledge base construction.

(4) Experiment results on datasets show that: (i) DSN can extract an average of 6.2 times more amount of knowledge from each text. (ii) the precision of knowledge we extract exceeds 64%.

The remainder of this paper is structured as follows: Section 2 describes some existing approaches for knowledge extraction and management in the literature. In Section 3, we introduce the scalable DSN architecture and the constructing in detail. In next Section, we describe the MapReduce framework for KBC. Then we present results of our experiments in Section 5. By comparing with other extraction results with different configurations on datasets, we show the effectiveness of our proposed model. In Section 6, we discuss the errors and weakness of our model. Finally, we conclude the paper in Section 7.

2. Related works

2.1. Data representation model

In traditional data presentation, bag-of-words (BOW) represent the data to be a vector space defined by the different terms. If a term occurs in the text, the corresponding value in the vector is nonzero. This kind of model only describes whether a term exists in the text, but cannot reflect the importance of this term for the text. To make up this shortcoming, a lot of weighting schemes [9,18,22,27,34] have been proposed, such as document frequency (TF) and the inverse document frequency (IDF) [35]. These schemes add appropriate weights to each dimension in vector space to indicate the importance of terms. However, BOW model has some unavoidable disadvantages. Particularly, its dimension is too large, and the relationship between each word vector cannot be described. Because of them, another way of data representation which focuses on neural models [14] was proposed. This kind of model can automatically extract dense features from textual data; typical methods include skip-gram (SG) [4], continuous bag-of-words (CBOW) [24], Paragraph Vector (PV) [19], and so on [10,17,20]. Usually, these models capture context information of textual terms.

Another way of data representation is a graph, which is divided into word order based graph and syntactic structure based graph, where the word order based graph model [1,21,28,38] can express the frequency of the word itself and the word order relation occurred in the text according to a specific window size. In some models, the nodes represent the terms, and the directed edges represent the order of words. The syntactic structure based graph model [33] analyzes the dependency between terms and shows the term's role that it plays inside the statement. In the others, the nodes represent the terms, and the edges represent the syntactic relation between the nodes.

Although vector-based representation methods and graph-based text representation methods can express the existing features of data to some degree, they have their respective shortcomings. For word vector, it can convert the text into a vector according to the similarity between the words, but cannot reflect the order relation of the words in the text and capture the role that the words play in the text. For graph model, it can express the order or structural feature of the statement on a single level, but cannot reflect the similarity of the text vocabulary and the semantic relations between the words, such as generalization and specialization.

2.2. Knowledge extraction and management

Knowledge base construction in PDC has been a curial and emerging research area recently. Along with the development of information processing technology, many creative and meaningful approaches emerge in past several years. Generally speaking, its work is divided into two categories: (1) facts from semi-structured data, such as YAGO [31], YAGO2 [16], DBpedia [3], and Freebase [6], which mainly comes from Wikipedia infoboxes and database; (2) facts from unstructured data, such as NELL [7], Reverb [12], OLLIE [30], Elementary [26], Prospera [25] and so on, which is extracted from web, text and document.

2.2.1. Knowledge from semi-structured data

The initial study focuses on the first category because there are a lot of semi-structured data for easy processing in the past. YAGO and DBpedia project extracts an ontological knowledge base from Wikipedia. DBpedia and YAGO have different class systems. While YAGO reuses WordNet and enriches it with the leaf categories from Wikipedia, the DBpedia project has manually developed its taxonomy. YAGO's compatibility with WordNet allows easy linkage and integration with other resources such as Universal WordNet, which we have exploited for YAGO2. For extracting relational facts from infoboxes, YAGO2 uses carefully handcrafted patterns, and reconciliations duplicate infobox attributes (such as birth date and date of birth), mapping them to the same canonical relation. DBpedia outsourced the task of pattern definition to its community and used a much larger number of more diverse extraction patterns, but ends up with redundancies and even inconsistencies. For semantic integration of heterogeneous XML data, data clustering and semantic-aware indexing can make some benefit from XML semantic analysis [32]. [36] proposes a knowledge base model by extracting useful information from social media.

2.2.2. Knowledge from unstructured data

With the explosive growth of data on the web, it is found that the coverage and real time of semi-structured data are not good enough. Researchers began to extract knowledge from the original data directly, which is the second category. Since the raw data is unstructured which is only in plain text with no or less structure information, it is necessary to use Natural Language Processing (NLP) technology. According to the different extraction methods of the processed data, this class can be divided into two types: rule-based systems and statistical approaches.

Reverb [12] uses simple syntactic and lexical constraints on binary relations expressed by verbs to matching open information. ReNoun [37] generalizes from the seed set to produce a set of extractions of noun attributes. These extracted semantic information will be mapped to relation triples in various ways, such as manually rules [30], co-occurring relations in a large distantly-labeled corpus [2], topic model [23], and clustering method [11].

In an open domain, it is difficult to ensure that the pattern set covers all situations and extracts with high accuracy. For the last decade, statistical approaches and machine learning have been proposed to select from a range of a priori features automatically. In these approaches, the extracted triple is associated with a marginal probability that it is true. NELL [7] uses logistic regression to select the most similar semantic shortest dependency patterns to update seed patterns in each iteration of bootstrapping. Microsoft's EntityCube [39] performs iterative distant supervision while using the ℓ_1-norm regularization technique to reduce noisy extraction patterns. Prospera [25] uses an approach based on constraint satisfaction to combines pattern-based relation extraction approaches with domain-knowledge rules and performs consistency checking against the existing YAGO knowledge base.

Besides, it is found that it is time-consuming and inefficient to build a large-scale knowledge base. Therefore, some researchers introduce distributed computing to the construction of knowledge base. Research on distributed knowledge extraction and distributed knowledge base construction begin and will become an area of intense study. Elementary [26] leverages high-throughput parallel computing infrastructure such as Hadoop, Condor, and parallel databases are used to support KBC with terabytes of data and millions of entities.

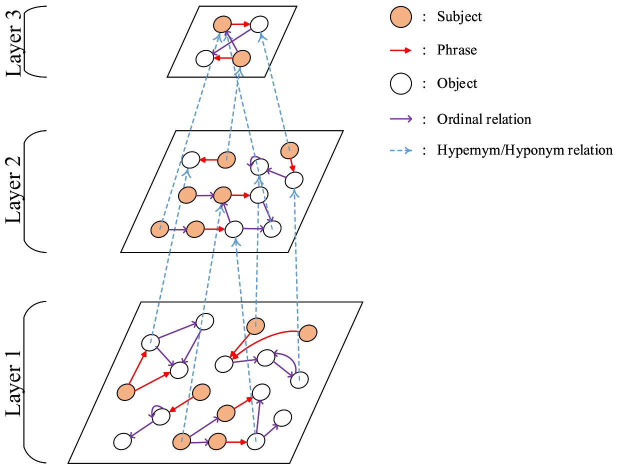


Fig. 1. The architecture of DSN with three concept layers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. Scalable distributed semantic network

The traditional data representation methods are not suitable for information extraction in distributed and parallel computing, and cannot retain sufficient semantic information, such as the order of terms and "hypernyms–hyponyms" relation of terms in texts. To solve the problem, we propose a hierarchical graph based DSN for representing semantic information extracted from heterogeneous data. Distributed semantics develops and studies theories and methods for quantifying and categorizing semantic similarities between linguistic items based on their distributional properties in large samples of language data. DSN can be used to describe the hierarchical and distributed semantics implied by text data. It not only expresses the <subject–predicate–object> relationship between text terms by phrases network but also shows the <generalization–specialization> relationship in a hierarchical structure. Therefore, DSN can absorb more distributed semantic features and be suitable for information extraction in distributed environments.

3.1. Architecture of distributed semantic network

The hierarchical architecture diagram of DSN is shown in Fig. 1. DSN is consist of several concept layers that connected by generalization and specialization relationship of the concept nodes in different layers. In each layer, the orange circles represent subject nodes, the white circles represent object nodes, the purple lines represent the ordering relationship between the nodes inside a subject or an object, the red lines represent predicates, and the blue lines represent the relation between hypernym and hyponym. The words in lower layers have more precise sense than words in higher layers, and words in high layers have broader sense than words in lower layers. The number of layers can be determined automatically by the sense of each concept. The sense of the concept can be obtained from the hierarchical relationships from WordNet, which is a large lexical database of English. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.

The distributed features of the model are revealed by the hierarchical structure, hierarchical relationships, the order and <subject–predicate–object> relations between nodes together. In DSN, all triples are distributed in different layers according to their semantic information. Inside each layer, all nodes are in different positions according to their roles they played in the statement.

3.2. Scalable distributed semantic network construction

CPS produces massive data from heterogeneous sources. When we have captured the data in different structures, DSN can be

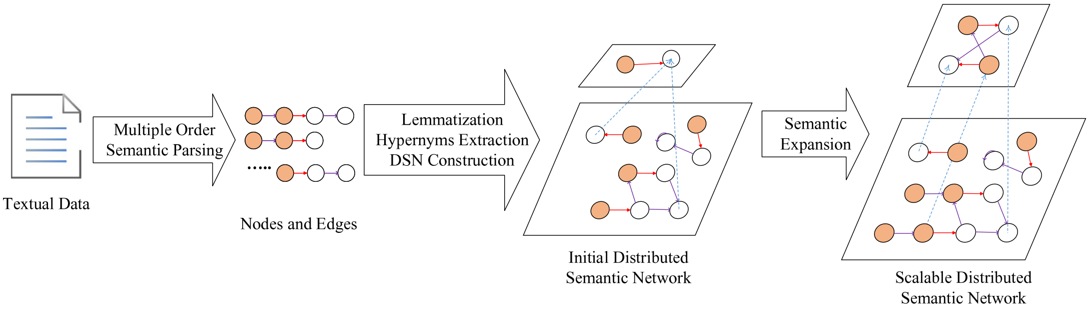

Textual Data
Multiple Order Semantic Parsing
Nodes and Edges
Lemmatization Hypernyms Extraction DSN Construction
Initial Distributed Semantic Network
Semantic Expansion
Scalable Distributed Semantic Network

Fig. 2. The overview of DSN construction process: extract concept nodes and relation edges by multiple order semantic parsing, lemmatization and hypernyms extraction for initial DSN construction, scalable DSN expansion by generalization–specialization.

constructed and scaled out by three steps: MOSP processing, DSN construction, and scalable DSN expansion. The overview of DSN construction process is shown in Fig. 2.

3.2.1. Concept entities and relationships extraction

We propose MOSP to preprocess data for extracting semantic information. The semantic information includes not only the concept entities and relationships in DSN, and also the generalization–specialization relationships between the concepts with different information entropy. Each statement in the data as input is turned to a triple in the form of <subject, predicate, object>. The "subject" and "object" are expressed as a set of nodes; the "predicate" is expressed as edges.

MOSP is a process of scattering and restructuring as shown in Fig. 3. Firstly, the input statements are preprocessed to remove the noisy information, such as structure information and some stop words, etc. Moreover, the statements are processed to split into fragments (words, phrases, symbols, and other meaningful segments) with semantic association information, which is completed in dependency parsing by the NLP Parsing tools. Next, these semantic elements are processed into a binary semantic structure, named Multiple Order Semantic Tree (MOS Tree). In some compound–complex sentence, there are often many clauses contained in the statement. We need to handle all clauses and merge the generated MOS Trees with the tree from the main clause. When we deal with all clauses, the whole MOS Tree would be optimized by the optimization process. Finally, a complete MOS Tree would be constructed.

(1) MOS tree construction

The root node of MOS Tree is essential at the beginning of the parsing. The relational phrase of the main clause is selected as the initial root node because it is also the root of the dependency path. Starting from the initial root node, we first find the possible *relational subject/object*. The corresponding patterns are learned from many MOS Tree examples and its dependency parsing in a corpus. The relational subject of a relational phrase may be the subject, possessive, relational phrase of nested clause, and so on, the relational object of a relational phrase may appear in some dependency structures such as the direct object, preposition structure, relational phrase of object clause, and so on. If the current root does not have a child node (subject, object or relational phrase), construction of semantic tree is completed. Otherwise, the child node will be a new root node for recursive construction.

For compound–complex statements, there is more than one clause. After construction of MOS Tree from the initial root, the next step is to find whether there is any other clause except the main clause (which is the origin of the MOS Tree from the initial root). For each "brother", a conjunction node will be generated as a parent node to connect the root of it and its brother. Moreover, the brother node will be a root for recursive construction. After the completion of construction of each clause and conjunction, we get a binary semantic tree of this statement.

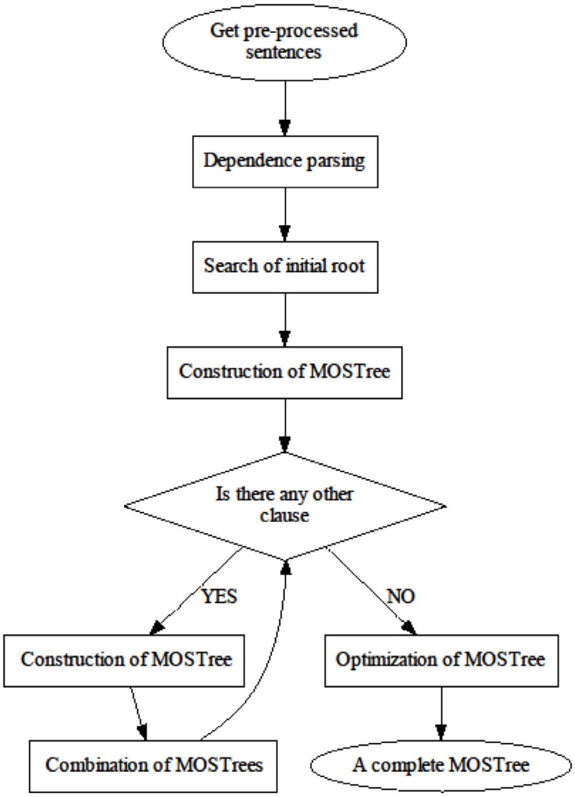


Fig. 3. The process of multiple order semantic parsing for concept entities and relationships extraction.

(2) MOS tree optimization

The semantic generating tree may contain single child subtree (only-leftchild or only-rightchild). Therefore, the following optimization work is to upgrade it into the strict binary tree.

$$T_{singlechild} = \begin{cases} T_{SPO}, \; T \; has \; binary \; semantic \\ Node, \; others. \end{cases}$$

The specific work depends on whether the subtree with complete semantic information. As showed above, different solutions to these two cases are designed to adjust the structure of the subtree. If it has a binary semantics, the subtree will be processed to a strict binary tree. Otherwise, semantic of these subtrees are processed to merge into a semantic object node as a "LeafNode". Finally, an MOS Tree will be constructed successfully, and each semantic will be obtained in the triple format by traversing the tree. Taking the statement in Fig. 4 as an example, the generating MOS Tree by this module is a four-layer structure. A total three pieces of semantic information in triple format can be obtained

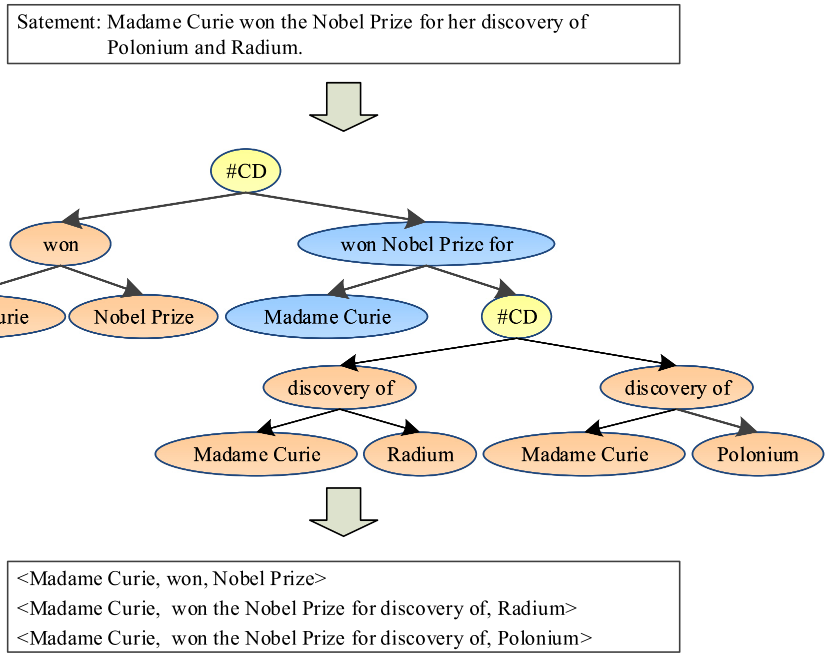

Satement: Madame Curie won the Nobel Prize for her discovery of Polonium and Radium.
#CD
won
won Nobel Prize for
Madame Curie
Nobel Prize
Madame Curie
#CD
discovery of
discovery of
Madame Curie
Radium
Madame Curie
Polonium
<Madame Curie, won, Nobel Prize>
<Madame Curie, won the Nobel Prize for discovery of, Radium>
<Madame Curie, won the Nobel Prize for discovery of, Polonium>

Fig. 4. An example of MOSP: extract three triples from the input statement of "*Madame Curie won the Nobel Prize for her discovery of Polonium and Radium*".

from this MOS Tree shown in Fig. 4. The component in triple information will be the nodes and edges needed in the DSN.

3.2.2. Initial distributed semantic network construction

The purpose of this section is to construct an initial DSN with hierarchical structure and a complete semantic information by combining the subject–predicate–object relation between terms and the relations between hypernyms and hyponyms. Based on the nodes and edges obtained in the previous step, we can construct an initial DSN through three steps: (1) lemmatization; (2) hypernyms extraction; (3) DSN construction.

In the previous section, we have obtained the nodes that are connected by the edges. Since the edges are all from predicates, they are usually verbs or verb phrases which have no hypernyms, so we can focus our attention on the nodes, and add the "hypernyms–hyponyms" relations between nodes into the DSN. To query the hypernym of a phrase in WordNet, we should firstly lemmatize the word. In this step, we traverse each node to reform the nouns contained in them, so that lay the foundation for the query of the hypernyms and merging of same nodes.

Now the words in the nodes can be queried in WordNet, but what we have to notice is that one node contains one word, while a subject or an object can be combined with more than one node. In this paper, we use three predefined rules to select a set of consecutive nodes as key words. The importance of each factor decreases in the sequence of length, part of speech (POS), and position.

– The length of the phrase: The whole phrase contains the most syntax information. We choose the phrase as long as possible. If we cannot find the hypernyms of the phrase consisting k words, we try $k - 1$ words next.

– POS of the word: The POS of a word can indicate different information entropy implied in it. Noun contains more syntax information in common sense, which followed by verb and adjective in sequence. We prefer to choose noun or adjective–noun phrase as a key word to query hypernyms. While the length is the preferential condition, POS is the second important factor.

– Position in the phrase: The latter position contains more syntax information. We prefer to choose latter words to query hypernyms. Position information is the lightest impactor.

After obtaining the hypernym–hyponym relations between the nodes, we can express the text according to the level of hypernyms, and hyponyms appeared in the WordNet tree structure. The number of layers in our model is decided by the number of layers in the WordNet tree structure that the datasets' vocabulary appeared, the relation between the layers is represented by the directed edge from hyponyms to hypernyms. Finally, we obtain the initial two layers DSN as shown in Fig. 5.

3.2.3. Scalable distributed semantic network expansion

After constructing the initial DSN, we can further use external information to do some semantic expansion for the model, then obtain the expanded DSN. The external information can be WordNet's synonym and hypernym information, as well as the relevant and category information extracted from Wikipedia. The process of semantic expansion is mainly divided into two parts: Horizontal expansion (using synonyms to expand subject and object) and vertical expansion (using hypernym to expand object). We describe the process in Fig. 6.

To scale out the initial DSN, we focus on the subject nodes and object nodes and do some expansion on them. First, we select a set of nodes that contain a subject or an object from the initial DSN. Here, we use object as an example. The first step is to acquire the object's key words by using the factors of length, POS, and position. We can get the whole object as key words in the beginning. If such key words cannot let the next steps work, then another key word is obtained until there are no key words remained. The second step is to input the key words into WordNet and Wikipedia to query its synonyms, if there exists a list of synonyms, we connect each of them to the previous words contained in the object to form a new object. Next, we input the new object into WordNet or Wikipedia and check if the new object is a right phrase. If it is not a right phrase, we cannot insert it into the model. In another word, if it is a right phrase, we insert it into the original model and connect it to the predicate. The method of handling hypernym is the same.

What is more, if there is a conjunction in an object, we should query the hypernyms of the words connected by the conjunction at the same time, and add the common hypernym of these several parallel words to the model. What has to be noticed is that this

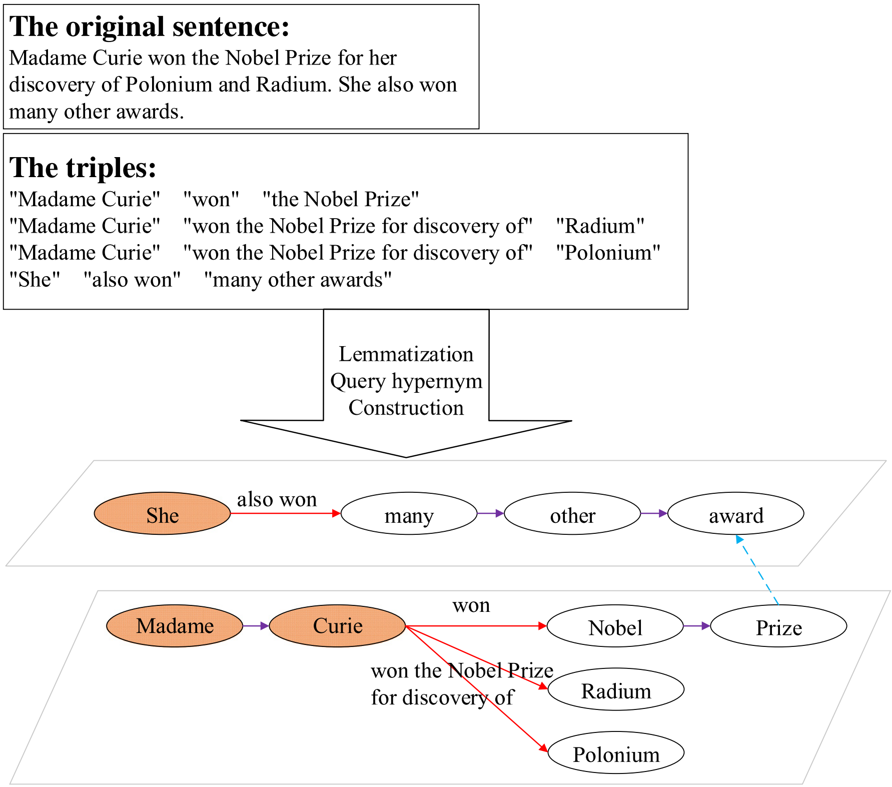

The original sentence:
Madame Curie won the Nobel Prize for her discovery of Polonium and Radium. She also won many other awards.
The triples:
"Madame Curie" "won" "the Nobel Prize"
"Madame Curie" "won the Nobel Prize for discovery of" "Radium"
"Madame Curie" "won the Nobel Prize for discovery of" "Polonium"
"She" "also won" "many other awards"
Lemmatization
Query hypernym
Construction
She
also won
many
other
award
Madame
Curie
won
Nobel
Prize
won the Nobel Prize
for discovery of
Radium
Polonium

Fig. 5. An example of initial two layers DSN construction.

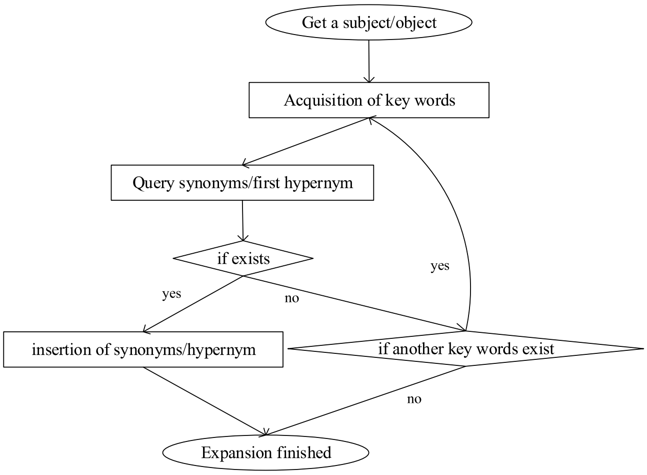

Get a subject/object
Acquisition of key words
Query synonyms/first hypernym
if exists
yes
no
yes
insertion of synonyms/hypernym
if another key words exist
no
Expansion finished

Fig. 6. The process of scalable DSN expansion using lexical database.

process may create many duplicated nodes and edges, we are better to merge the same nodes and edges in this step to simplify the model's complexity. Finally, the scalable DSN is acquired.

4. MapReduce-based knowledge base construction

Distributed and parallel framework is a method to solve the problem of quickly constructing a large knowledge base. Because of a structural feature of our proposed DSN, it is convenient for distributed computing. After constructing the scalable DSN from the unstructured text by the method described in the previous section, the following work is to use the extracted and expanded semantic information to build the knowledge base. This section is divided into two parts: scalable DSN based knowledge management and MapReduce based framework architecture.

4.1. Scalable DSN based knowledge management

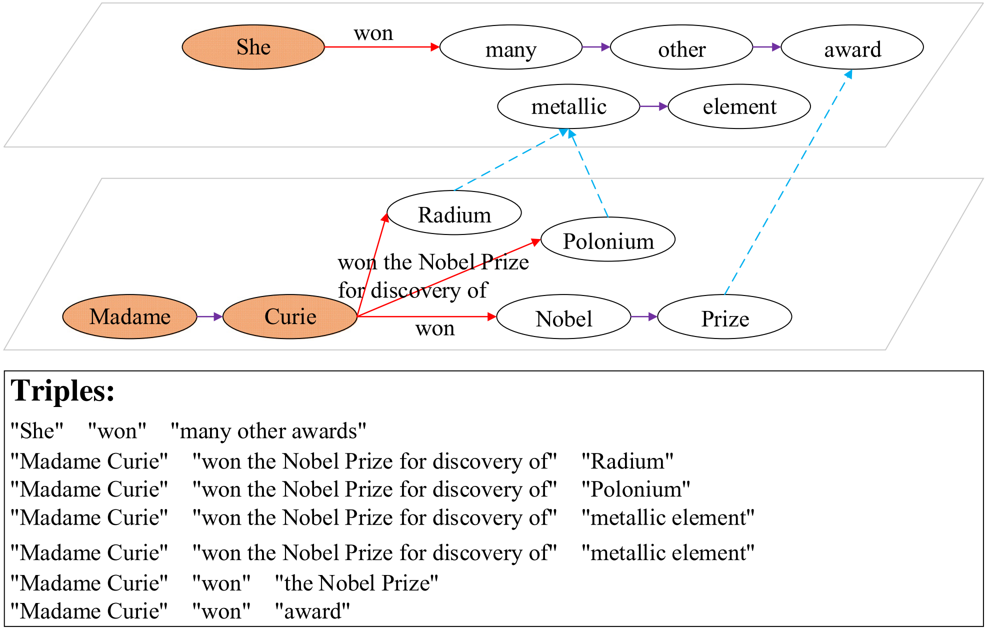
Based on the scalable DSN constructed in the previous section, we need to extract semantic information from it and refine the knowledge by data fusion process.

4.1.1. Knowledge extraction from scalable DSN

Knowledge extraction, the basic of knowledge base construction, is to directly obtain the semantic information from the DSN in the form of triples. DSN is composed of all the original information and expanded information from the raw text. Each path in DSN represents one or more pieces of semantic information. Therefore, the knowledge extraction from DTSM is a traversal of the networks in essence.

The traversal process starts from the starting node (without precursor), which is the first word of the subject in the triple. Along the directed edge in each layer of DSN, we read the semantic information until ending node (without successor) to form a semantic path. As mentioned before, the generated semantic path may consist of more than one piece of semantic information. We will get each piece of semantic information by splitting the path into multiple semantic sub paths. By dividing the entity pairs and their relationship from these semantic paths, the semantic information is reorganized into triple formatted knowledge. In particular, in the process of traversing the DSN, the relation between associated nodes in neighboring layer implies expansion semantic, which need to be extracted separately. By such method, we can get all of the possible triples from the DSN.

An example of knowledge extraction is shown in Fig. 7. It can be learned that there are two layers with 12 nodes in the DSN. Ten nodes are obtained from the text, while the nodes "metallic" and "element" are expanded nodes by its hypernym relation of



Triples:

"She" "won" "many other awards"
"Madame Curie" "won the Nobel Prize for discovery of" "Radium"
"Madame Curie" "won the Nobel Prize for discovery of" "Polonium"
"Madame Curie" "won the Nobel Prize for discovery of" "metallic element"
"Madame Curie" "won the Nobel Prize for discovery of" "metallic element"
"Madame Curie" "won" "the Nobel Prize"
"Madame Curie" "won" "award"

Fig. 7. An example of knowledge extraction from two layers DSN.

"Radium" and "Polonium". In the process of extraction, we start from the node "She" and "Madame", and traverse all of the paths starting from them. It can be seen that there are four paths in Fig. 7 as follows:

1. She—$^{\text{won}}$ → many → other → award
2. Madame → Curie—$^{\text{won the Nobel Prize for discovery of}}$ → Radium—$^{\text{hypernym}}$ → metallic → element
3. Madame → Curie—$^{\text{won the Nobel Prize for discovery of}}$ → Polonium—$^{\text{hypernym}}$ → metallic → element
4. Madame → Curie—$^{\text{won}}$ → Nobel → Prize—$^{\text{hypernym}}$ → award.

The nodes of these four semantic paths consist of words of entities, and the edges with words express the relation. From the four paths, we can get seven triples as shown in Fig. 7.

4.1.2. Knowledge refinement by information fusion

After the knowledge in triple being obtained, it can be found that information may be messy, duplicated or even conflicting. Duplicated and conflicting information will affect the accuracy of the knowledge base, resulting in fuzzy knowledge, and even reduce the unreliability of the knowledge base.

Information fusion is the process of integrating knowledge extracted from DSN to produce more consistent, accurate, and useful knowledge than that provided by the raw text. We design an information fusion method to complete knowledge refinement. This method focuses on the Polysemy and Synonyms problem by clustering. All pieces of knowledge are clustered into a few groups, which is used to detect the repeated knowledge and refine the semantic relation with different phrases. With the help of refinement, knowledge is processed into a standard format. Besides, heterogeneous sources of data may bring conflicting information. To address this issue, we use statistical inference to combine different semantic information as several recent KBC projects do. Associated with each such triple is a confidence score, representing the probability that the triple is considered correct. In the previous section, we get four semantic paths and seven triples. The triples are refined into six different triples by abandoning a duplicate triple <"Madame Curie", "won the Nobel Prize for the discovery of", "metallic element">.

4.2. MapReduce based PDC framework

With the help of Scalable DSN, we easily use distributed parallel processing technique in information extraction and fusion to provide scalable and efficient knowledge base construction. MapReduce is a software framework that can be used for processing and generating large datasets. Users specify a map function that splits data into key/value pairs, and a reduce function that merges all key/value pairs based on the key. The MapReduce framework is easily parallelizable for execution on large clusters of commodity machines. That is helpful for the construction of high-performance, highly-scalable applications.

The workflow of MapReduce-based knowledge base construction is shown in Fig. 8. The input of the workflow is the DSN, and the output is the target knowledge base. The framework mainly consists of five steps: Data partition, Map, Shuffling, Reduce, and Construction.

4.2.1. Data partition

In the MapReduce framework, data partition is to split input data into several data blocks for the following processing. However, our proposed model is different from the general processing data since connected edges of one node cannot be separated to process. The nodes connected in the DSN are represented within the same layer, and only the hierarchical information can cross different layers. The DSN can be divided into some layers, which include complete semantics and are small as far as possible.

Since our DSN is stored with the list of nodes per layer with an adjacency list, it is easy to partition the model by line. After dividing the model into multiple layers, we check whether there is a connection with the upper layer. If there were, we would copy the subsequent information to combine with the semantic information of this layer into a group. After dividing the model into multiple groups, each group contains a set of nodes and their directed edges, which formed a subgraph containing independent semantic information. Then this information will be transferred to Mappers for map task. For example, the representation graph in Fig. 7 can be split into two groups. One consists of the nodes and edges in the upper layer and includes the concepts of more information entropy. The lower layer is made up of the nodes and edges with associated "metallic" and "element" and includes the concepts more concrete.

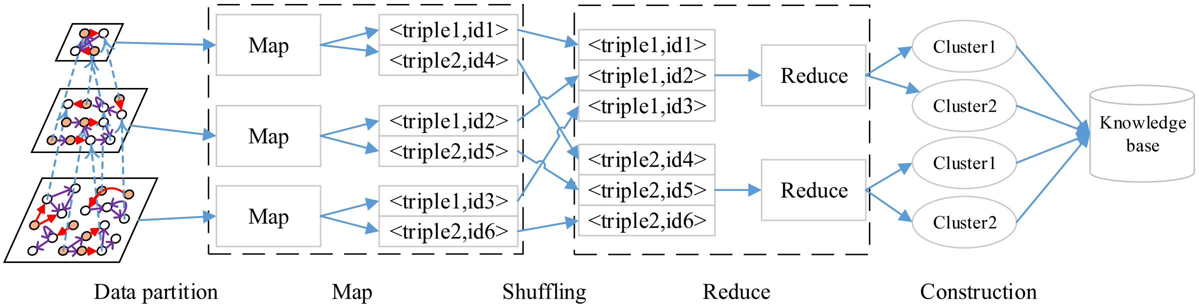


Fig. 8. The workflow of MapReduce based knowledge base construction framework.

4.2.2. *Map*

The Map task in our framework is to map the input data into the triple format. Once assigned to the Map task and input graphs, Mapper executes the map function to analyze graphs for extracting the semantic information as described in the previous section. The extraction processing in Map function is shown in Algorithm 1. In MapReduce based framework, changing a task's memory requirement requires changing the following parameters: (1) The size of the container in which the map/reduce task is launched. (2) Specifying the maximum memory (-Xmx) to the JVM of the map/reduce task. For the Map function in Algorithm 1, the two parameters (mentioned above) are changed for MapReduce in Hadoop 2 as shown below:

```
mapreduce.map.memory.mb=2240   # Container size
mapreduce.map.java.opts=-Xmx2016m  # JVM arguments
```

```
Algorithm 1.
Function Map (String key, String value)
Input: id-adjacency list
Output: triple-id
$R_v$ = FindRoot(value);
foreach (r ∈ $R_k$) do
    p = DFS(r)
    if(p contains semantic information)
        $P_v$.add(p)
foreach (p ∈$P_v$) do
    $TF_v$ = p.split
    foreach (tf ∈$TF_v$) do
        t = generate triple from tf
        EMIT-INTERMEDIATE(t, key)
    $TF_i$ = inference($TF_v$)
    foreach (tfi ∈ $TF_i$) do
        $t_i$ = generate triple from $tf_i$
        EMIT-INTERMEDIATE($t_i$, key)
```

The input of Map includes id and the adjacency list of a group, which is generated by data partition. The output of Map, called intermediate results, is the key–value pairs, in which the key is the semantic information in a triple format and the value is the id. The Map is the specific implementation of the process of information extraction described in the previous section. First of all, we find all the starting node that does not have a precursor in the graph of the group. Each starting node represents a semantic path, which contains one or more semantic information. Then, we use the starting nodes as the root to perform depth first traversal of the input graphs to get the paths. In this step, some paths which do not contain complete semantic information will be abandoned. If there is more than one starting node, each starting node is performed the same operation to extract all semantic triples. Next, the semantic information in triple format is obtained by reorganization of these paths. These paths are split into multiple fragments by the semantic edges that express the semantic relationships. A complete semantic information can be obtained from each fragment. Moreover, expanded semantic information can be reasoned by special semantic edges. Finally, this semantic information is organized into triples as the key of the key–value pair for the next step.

4.2.3. *Shuffle*

When all Map tasks have completed, the intermediate results should be shuffled and assigned to reducers. The mapped outputs would be sorted and merged by intermediate keys, i.e. key2, so that the result is stored with reorganization in local disks and all values of the same key are grouped. At the same time, partition calculates that the key–value pairs should be handed over to which reducer for next processing. Once all mapped outputs are already partitioned and stored in local disks, each reducer performs the shuffling by simply pulling its partition of the mapped outputs from mappers. Each record of the mapped outputs is assigned to only a single reducer by one-to-one shuffling strategy.

4.2.4. *Reduce*

The Reduce task in our framework is to refine the semantic triples. Once assigned to the input triples, Reducer read the intermediate result from Mappers to remove the redundancy from intermediate result. Reduce function, a part of information fusion, is described in Algorithm 2. The two parameters are changed for MapReduce in Hadoop 2 as shown below:

```
mapreduce.reduce.memory.mb=2240   # Container size
mapreduce.reduce.java.opts=-Xmx2016m  # JVM arguments
```

```
Algorithm 2.
Function Reduce (String key, Iterator values)
Input: triple-id list
Output: triple
if (there is a knowledge base K)
    if(!K.contains(key))
        EMIT(t)
else
        EMIT(t)
```

It can be seen Reduce task is relatively straightforward. The input of reduce task is the key–values pair, which is generated by shuffling from mapper. The output of reduce task is semantic triple without redundancy. The values in input pair are the list of value that associated with the same key, which is done by Shuffling to make easy to dedupe. Whenever an input pair is obtained, Reducer abandons the value while preserving semantic triples in the key. These semantic triples are then used to build the knowledge base. In particular, if there is already an existing knowledge base to expand, not only to detect the repeated extract information inside but also to identify the repeated extract information with the knowledge base as shown above.

4.2.5. Construction

When all the Map and Reduce tasks are performed, each Reducer's output is in its corresponding output file. Then, these outputs are going to start the second screening by refinement and statistical inference. With the help of previous studies, we use the similarity computation to group the triples with approximate semantic into a cluster. To measure a better similarity of triples, A novel similarity measure, denoted by maxsim(t_i, t_j), is designed to capture the degree of similarity between any two relation triples t_i, $t_{j,}$. The measure is defined as:

$$\text{maxsim}\left(t_i, t_j\right) = wn\left(t_i, t_j\right) \times con\left(t_i, t_j\right)$$

where $wn(t_i, t_j)$ is the similarity of the relation triples in literal meaning to deal with the synonym problem. In this step, we use a semantic word similarity model constructed by combining Latent Semantic Analysis (LSA) based word similarity and WordNet knowledge to get a similarity score between relational phrases of 0 to 1 range [15].

After obtaining the similarity of the relation phrase, these phrases need to be grouped into one cluster that can then be treated as one collective unit. We choose the Markov Clustering (MCL) to achieve this clustering work in our system. MCL decomposes the word graph into small coherent pieces via simulation of random walks in the graph that eventually get trapped in dense regions, the resulting clusters. The center will refine each triple in the same cluster. After the clustering, the semantic information is deduped again. Besides, we use the network to calculate the confidence of semantic information, which is used for statistical inference to resolve semantic conflicts. Finally, a more complete and powerful knowledge base will be generated.

5. Experiments

5.1. Data and evaluation

We propose scalable DSN and a distributed knowledge base construction framework to extract and manage knowledge. To validate that DSN is suitable and helpful for distributed computing, we construct the DSN from several unstructured data to generate the knowledge base by MapReduce based framework. For the generated knowledge base evaluation, we use heterogeneous datasets from different sources, consisting of three textual data, is presented and manually labeled in ClausIE [8]. The details of the datasets are as follows:

(1) Wiki dataset: it contains 200 random short, simple, and less noisy statements from Wikipedia pages.

(2) New York Times dataset: it is a collection of 200 statements from the New York Times (NYT) collection. These statements are very clean but tend to be long and complex.

(3) REVERB dataset: it consists of 500 statements obtained via the random-link service of Yahoo and is very noisy.

Different from the existing systems, our proposed framework is applicable for heterogeneous datasets. The three datasets used in experiments include text data with different formats and structures. Wiki dataset includes not only text but also some web structures. NYT includes a large number of complex and compound statements with no structures. REVERB is a subset of ReVerb's output run on the English portion of the ClueWeb09 corpus, and the file has tab-separated columns strictly.

Table 1
The precision of different systems on each dataset.

	NYT	Wiki	Reverb
REVERB	54.98	**66.27**	53.37
OLLIE	42.45	41.42	44.04
ClausIE	53.42	59.74	57.34
S-DSN	**69.27**	64.77	**65.58**

Table 2
The number of correct extractions and total extractions.

	NYT	Wiki	Reverb
REVERB	149/271	165/249	388/727
OLLIE	211/497	234/565	547/1242
ClausIE	696/1303	598/1001	1706/2975
DSN	542/749	540/765	1297/1785
S-DSN	924/1334	796/1229	1972/3007

To evaluate the effect of our proposed framework, we compare with other systems by computing precision and an absolute number of knowledge on datasets. We use the absolute number of extractions instead of recall because of the reasons that it is infeasible to obtain the set of "all correct" propositions proposed by Luciano et al. We automatically evaluate the fact of initially labeled triples and then evaluate the extended additional knowledge manually. The expansion of original false information is considered wrong. Besides, rest expansions were labeled by two independent labelers. Like ClausIE, an expansion was treated as correct only if it was labeled as correct by both labelers.

5.2. Experimental results

Our proposed method is compared against previous works on information extraction to verify its effectiveness on three different datasets. We compare our method to three state-of-the-art representative information extraction systems (REVERB [12], OLLIE [30] and ClausIE [8]) as described in [5,29]. We use precision to evaluate the result of semantic information extraction from heterogeneous data included in the three datasets.

$$Precision = \frac{number\ of\ correct\ extractions}{number\ of\ total\, extractions}.$$

The precision of the three systems and our proposed method are presented in Table 1. "S-DSN" means that the results are obtained from scalable DSN. From the precision value of each system on different datasets, we can find that in addition to a slight gap with REVERB on Wiki dataset, S-DSN has got the best precision among all the different methods on NYT dataset and Reverb dataset. Also, the average precision of our method is over 65%, which is much higher than other methods. That is because S-DSN only chooses reliable rules in MOSP and make a verification of expansion. It is not worthwhile to lose a large amount of correctness for a small number of extractions. Moreover, later we will continue to improve our method to increase the precision under the premise of not reducing or even increasing the number of correct extraction.

Next, we look at the absolute amount of extracted knowledge. Table 2 presents the number by different methods on each dataset. The figures in the row of "DSN" are the number of directly obtained information by MOSP from statements. For each method and dataset, we use two different numbers: the *number of correct extractions/number of total extractions*.

As shown in Table 2, S-DSN not only has a higher precision but also obtains both the maximum number of correct information and a total number of results than three other methods. Although the number of information directly extracted from statements

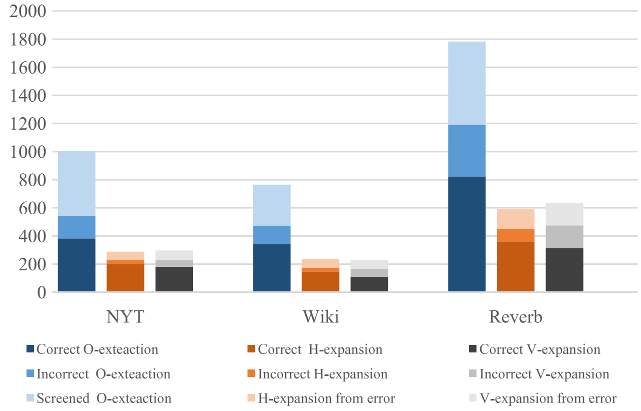


Fig. 9. The detailed number of expansion on different datasets. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is slightly less than ClausIE, we conduct a more accurate, non-redundant extraction, and finally get a greater number by expansion as shown in the last row.

From the data in Table 2, we can find that expansion on NYT dataset increases 585 (394 correct) pieces of information, expansion on Wiki dataset increases 464 (252 correct) pieces of information, and expansion on Reverb dataset increases 1250 (672 correct) pieces of information. The numbers of expansion information on the three datasets reach 60% of the number of direct extraction information. We can conclude that the expansion can increase the number of extracted knowledge. That is because we added extra semantic information to the expansion process and got some more inferential information for knowledge base construction.

For a more detailed analysis of the effect of the expansion, we compare the results of horizontal expansion and vertical expansion on each dataset. Fig. 9 shows the detailed data of expansion. The blue bar represents the number of the original triples before expansion, the orange bar stands for the number of the correct horizontal expansion, and the green bar describes the number of the correct vertical expansion. Also, the number of original information includes all the number of the correct, wrong and screened information described in Section 3.2. The number of horizontal expansion and vertical expansion consist of the number of correct expansion, incorrect expansion from correct original information and the number of expansion from incorrect original information.

It is shown that the expansion greatly increases the number of semantic information. As shown in Fig. 9, compared to the initial screening data, the horizontal expansion increases the number of information respectively by 53.04% (288) on NYT, 49.58% (235) on Wiki and 51.89% (618) on Reverb. Meanwhile, the vertical expansion increases the number of information respectively by 54.70% (297) on NYT, 48.51% (229) on Wiki, and 53.32% (635) on Reverb. The data in brackets is the proportion of the expansion information and original information. Horizontal expansion and vertical expansion respectively increase the number of information by over 48% than original filtered information on each dataset. Although the initial information contains a certain amount of error information, the results reach a higher precision. It can be obtained from Fig. 8 that the precision of horizontal expansion on three datasets is: 69.79% (87.77%) on NYT, 61.70% (82.68%) on Wiki, and 61.19% (80.22%) on Reverb. And the precisions of vertical expansion are: 60.94% (79.74%) on NYT, 48.47% (67.27%) on Wiki, and 49.45% (66.11%) on Reverb. The data in brackets is the precision of expansion without an extension from incorrect original information. It can be seen that the precision of expansion from correct information is not low. However, the initial information is not completely correct. As mentioned before, the expansion of incorrect information is all considered incorrect so that the result is not very high. Meanwhile, the precision of horizontal expansion is higher than vertical expansion because that the substitution of synonyms has less impact on semantics than hypernym–hyponym.

Table 3
Time consumption of knowledge base construction by different implementation and different datasets (seconds).

	NYT	Wiki	Reverb
Serial implement	153	205	711
MapReduce-based	26	30	62

Finally, we evaluate the improvement of performance by distributed computing. We use Hadoop 2 as experiment platform. MapReduce based experiment is deployed on 4 PCs with 3.30 GHz dual-core CPU, 6 G RAM, and 100 G hard disk. Table 3 shows the performance of proposed method based on the serial implement and distributed implement. It is shown that MapReduce-based framework greatly improved the performance on each dataset. On the three datasets, our proposed framework shortens the process time by 5.8 times(NYT), 6.8 times (Wiki) and 11.5 times (Reverb). It can be seen that with the increase of data scale, the performance of distributed framework is more obvious. The first two datasets are small in scale and less obvious in the promotion. That is due to more layers of DSN generated from larger scale data. It can be divided into more partitions to distributed computing. Besides, there are some non-distributed parts in the whole process, so the performance of simple distributed parts increases higher.

6. Discussion

In this section, we analyze the errors and omissions in our proposed method. At first, we summarize the rules of incorrect extraction and expansion in the results. Through the comparison of the statements with their labeled information, we find following errors:

(1) Parse error. Some statements, which is in the bad grammatical forms or the complex structure with many special symbols, can lead to incorrect dependency parse and multiple order semantic parsing in the first step. This error affects the following extraction and expansion.

(2) Incorrect Expansion. In addition to the error expansion from the incorrect original information, there is a part of semantic caused by expansion. Not all relations can be expanded, and not all synonyms and the hypernym–hyponym can be replaced under correct semantic in a particular relation.

Next, we also analyze the labeled information that we did not get directly from MOSP. In addition to the parse error described above, it is found that two types of information are ignored. One is the redundant information, which is a combination of multiple pieces of information. Our method gets them respectively. Another is non-binary relation information. Our goal is to construct knowledge base that mainly consists of two elements. Non-binary relation information is often the attribute information and meaningless information, which is ignored in our method.

Besides, our MapReduce framework is not perfect that the refinement and conflict handling are simple. These steps take much time, which decreases overall performance. In the future work, we will start the research about these two aspects to improve the methods of our framework. Also, we will continue to research knowledge reasoning on the implicit semantic information.

7. Conclusion

We presented scalable DSN for express distributed semantics implied in data and semantic information extraction. Based on the extracted information, a knowledge parallel extraction framework with MapReduce for knowledge base construction from heterogeneous data was proposed. It first deals with heterogeneous textual data by MOSP to extract the semantic information, and then uses the extracted semantic information and existing semantic database (WordNet and Wikipedia) to construct the DSN and expose implicit semantic information by horizontal expansion and vertical expansion. Finally, all extracted and fused semantic information was used to construct the knowledge base based on MapReduce PDC framework. Experimental results show that our method outperforms the three state-of-the-art systems both on precision and the amount of correct knowledge. The PDC framework can greatly improve the efficiency of knowledge base construction.

The next step in our research is to adjust the PDC framework for performance improvement. There is also some target relational information not directly resolved in MOSP, which requires our system not only to improve the relevant rules for semantic information but also to cover semantic information on the non-standard statements as much as possible.

References

[1] C. Aggarwal, P. Zhao, Graphical models for text: a new paradigm for text representation and processing, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2010, pp. 899–900.

[2] G. Angeli, M.J. Premkumar, C.D. Manning, Leveraging linguistic structure for open domain information extraction, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, ACL 2015, 2015.

[3] S. Auer, C. Bizer, G. Kobilarov, et al., DBpedia: A nucleus for a web of open data, Semant. Web (2007) 722–735.

[4] Y. Bengio, R. Ducharme, P. Vincent, et al., A neural probabilistic language model, J. Mach. Learn. Res. 3 (Feb) (2003) 1137–1155.

[5] N. Bhutani, H.V. Jagadish, D.R. Radev, Nested propositions in open information extraction, in: EMNLP, 2016, pp. 55–64.

[6] K. Bollacker, C. Evans, P. Paritosh, et al., Freebase: a collaboratively created graph database for structuring human knowledge, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, ACM, 2008, pp. 1247–1250.

[7] A. Carlson, J. Betteridge, B. Kisiel, et al. Toward an architecture for never-ending language learning, in: AAAI, Vol. 5, 2010, p. 3.

[8] L. Del Corro, R. Gemulla, Clausie: clause-based open information extraction, in: Proceedings of the 22nd International Conference on World Wide Web, ACM, 2013, pp. 355–366.

[9] Z.H. Deng, K.H. Luo, H.L. Yu, A study of supervised term weighting scheme for sentiment analysis, Expert Syst. Appl. 41 (7) (2014) 3506–3513.

[10] M. Denil, A. Demiraj, N. Kalchbrenner, et al. Modelling, visualising and summarising documents with a single convolutional neural network. arXiv preprint arXiv:1406.3830, 2014.

[11] A. Dutta, C. Meilicke, H. Stuckenschmidt, Enriching structured knowledge with open information, in: Proceedings of the 24th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2015, pp. 267–277.

[12] A. Fader, S. Soderl, O. Etzioni, Identifying relations for open information extraction, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2011, pp. 1535–1545.

[13] D. Ferrucci, E. Brown, J. Chu-Carroll, et al., Building Watson: An overview of the DeepQA project, AI Mag. 31 (3) (2010) 59–79.

[14] Y. Goldberg, A primer on neural network models for natural language processing, J. Artificial Intelligence Res. (JAIR) 57 (2016) 345–420.

[15] L. Han, A.L. Kashyap, T. Finin, et al. UMBC_EBIQUITY-CORE: Semantic textual similarity systems, in: SEM@ NAACL-HLT, 2013, pp. 44–52.

[16] J. Hoffart, F.M. Suchanek, K. Berberich, et al., YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia, Artificial Intelligence 194 (2013) 28–61.

[17] M. Iyyer, V. Manjunatha, J.L. Boyd-Graber, et al. Deep unordered composition rivals syntactic methods for text classification, in: ACL (1), 2015, pp. 1681–1691.

[18] Y. Kim, O. Zhang, Credibility adjusted term frequency: A supervised term weighting scheme for sentiment analysis and text classification. arXiv preprint arXiv:1405.3518, 2014.

[19] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31st International Conference on Machine Learning, ICML-14, 2014, pp. 1188–1196.

[20] J. Li, Feature weight tuning for recursive neural networks. arXiv preprint arXiv:1412.3714, 2014.

[21] F.D. Malliaros, K. Skianis, Graph-based term weighting for text categorization, in: 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, (ASONAM), IEEE, 2015:, pp. 1473–1479.

[22] J. Martineau, T. Finin, Delta TFIDF: An improved feature space for sentiment analysis, ICWSM 9 (2009) 106.

[23] O. Melamud, J. Berant, I. Dagan, et al. A two level model for context sensitive inference rules, in: ACL (1). 2013, pp. 1331–1340.

[24] T. Mikolov, K. Chen, G. Corrado, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[25] N. Nakashole, M. Theobald, G. Weikum, Scalable knowledge harvesting with high precision and high recall, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, ACM, 2011, pp. 227–236.

[26] F. Niu, C. Zhang, C. Ré, et al., Elementary: Large-scale knowledge-base construction via machine learning and statistical inference, Int. J. Semant. Web Inf. Syst. (IJSWIS) 8 (3) (2012) 42–73.

[27] G. Paltoglou, M. Thelwall, A study of information retrieval weighting schemes for sentiment analysis, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 1386–1395.

[28] L. Ramachandran, E.F. Gehringer, A word-order based graph representation for relevance identification, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, ACM, 2012, pp. 2327–2330.

[29] J.M. Rodríguez, H.D. Merlino, P. Pesado, et al., Performance evaluation of knowledge extraction methods, in: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, Springer International Publishing, 2016, pp. 16–22.

[30] M. Schmitz, R. Bart, S. Soderl, et al., Open language learning for information extraction, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics, 2012, pp. 523–534.

[31] F.M. Suchanek, G. Kasneci, G. Weikum, Yago: a core of semantic knowledge, in: Proceedings of the 16th International Conference on World Wide Web, ACM, 2007, pp. 697–706.

[32] J. Tekli, An overview on XML semantic disambiguation from unstructured text to semi-structured data: Background, applications, and ongoing challenges, IEEE Trans. Knowl. Data Eng. 28 (6) (2016) 1383–1407.

[33] G. Veena, S. Krishnan, A concept based graph model for document representation using coreference resolution, in: Intelligent Systems Technologies and Applications, Springer, Cham, 2016, pp. 367–379.

[34] S.I. Wang, C.D. Manning, Baselines and bigrams: Simple, good sentiment and topic classification, in: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers, Vol. 2, 2012, pp. 90–94.

[35] H.C. Wu, R.W.P. Luk, K.F. Wong, et al., Interpreting tf-idf term weights as making relevance decisions, ACM Trans. Inf. Syst. (TOIS) 26 (3) (2008) 13.

[36] Z. Xu, H. Zhang, C. Hu, et al., Building knowledge base of urban emergency events based on crowdsourcing of social media, Concurr. Comput.: Pract. Exp. 28 (15) (2016) 4038–4052.

[37] M. Yahya, S. Whang, R. Gupta, et al. ReNoun: Fact extraction for nominal attributes, in: EMNLP, 2014, pp. 325–335.

[38] F. Zhou, F. Zhang, B. Yang, Graph-based text representation model and its realization, in: 2010 International Conference on Natural Language Processing and Knowledge Engineering, (NLP-KE), IEEE, 2010, pp. 1–8.

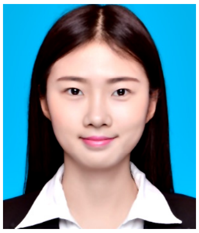
[39] J. Zhu, Z. Nie, X. Liu, et al., StatSnowball: a statistical approach to extracting entity relationships, in: Proceedings of the 18th International Conference on World Wide Web, ACM, 2009, pp. 101–110.

Shengli Song is currently an associate professor with Xidian University. He received his Ph.D. degree in Technology of Computer Application from Xidian University, Xi'an, China, in 2011. His current research interests include semantic computing and distributed system.

Yishuai Lin is currently a lecturer with Xidian University, China. She received her Ph.D. degree in Computer Science from Université de Technologie de Belfort-Montbéliard, Belfort, France, in 2013. Her current research interests include domain knowledge model, ontology and multi-agent system.

Bin Guo is currently a professor with Northwestern Polytechnical University, China. He received his Ph.D. degree in computer science from Keio University, Tokyo, Japan, in 2009. His current research interests include ubiquitous computing, social and community Intelligence, mobile crowd sensing and human–computer interaction.

Qiang Di is currently a master student majoring in Computer Science and Technology in Xidian University. His research interest is information extraction.



Rong Lv is currently a master student majoring in Computer Science and Technology in Xidian University. Her research interest is text representation.