Front. Comput. Sci., 2018, 12(1): 55–74
https://doi.org/10.1007/s11704-016-5228-9

REVIEW ARTICLE

A retrospective of knowledge graphs

Jihong YAN[1,2], Chengyu WANG[1], Wenliang CHENG[1], Ming GAO (✉)[1], Aoying ZHOU[3]

1 Institute for Data Science and Engineering, East China Normal University, Shanghai 200062, China
2 Institute for Computer and Information Engineering, Shanghai Second Polytechnic University, Shanghai 201209, China
3 Shanghai Key Lab for Trustworthy Computing, East China Normal University, Shanghai 200062, China

Abstract Information on the Internet is fragmented and presented in different data sources, which makes automatic knowledge harvesting and understanding formidable for machines, and even for humans. Knowledge graphs have become prevalent in both of industry and academic circles these years, to be one of the most efficient and effective knowledge integration approaches. Techniques for knowledge graph construction can mine information from either structured, semi-structured, or even unstructured data sources, and finally integrate the information into knowledge, represented in a graph. Furthermore, knowledge graph is able to organize information in an easy-to-maintain, easy-to-understand and easy-to-use manner.

In this paper, we give a summarization of techniques for constructing knowledge graphs. We review the existing knowledge graph systems developed by both academia and industry. We discuss in detail about the process of building knowledge graphs, and survey state-of-the-art techniques for automatic knowledge graph checking and expansion via logical inferring and reasoning. We also review the issues of graph data management by introducing the knowledge data models and graph databases, especially from a NoSQL point of view. Finally, we overview current knowledge graph systems and discuss the future research directions.

Keywords knowledge graph, knowledge base, information extraction, logical reasoning, graph database

Received June 11, 2015; accepted January 19, 2016

E-mail: mgao@sei.ecnu.edu.cn

1 Introduction

The central task of knowledge engineering is to extract the useful information from data, then to integrate the piecewise information into comprehensive, well-organized knowledge. Knowledge graph provides a general framework to represent knowledge, based on the mining and analysis of entities and relationships.

A knowledge graph (KG) is a semantic graph consisting of vertices (or nodes) and edges. The vertices represent concepts or entities. A concept refers to the general categories of objects, such as scientist, car, etc. An entity is a physical object in the real world such as a person (e.g., Micheal Jordan), a location (e.g., New York) and an organization (e.g., United Nations). The edges represent the semantic relationships between concepts or entities. Leveraging on KG, the fragmented, hence partially observed entities and concepts can be connected together to form a complete and structured knowledge repository, facilitating the management, retrieval, usage and understanding of the information it contains.

Knowledge base (KB) [1] is a similar approach to KG, which refers to an intelligent database for managing, storing and retrieving complex structured and unstructured information. In some cases, there are not many distinctions between KG and KB. KB emphasizes on the knowledge stored in databases, while KG focuses more on graphical structures. They do have a lot of features in common. In this survey, we use the two concepts interchangeably in the following sections.

The employment of KGs is a reasonable example in this big data era, driven by both the industry requirements and research motivations from academia. Nowadays, all the major search engines, such as Google and Bing, have employed KGs to provide dedicated answers to the search queries, such as “wife of Obama” where the query can be answered directly by entities, instead of a collection of relevant Web documents which contain a lot of redundant information.

KGs are able to infer the conceptual meanings of queries and identify user’s tasks in Web search [2], which can lead to better query suggestions. The knowledge in KGs enables machines to understand natural language text [3] and semi-structured Web tables [4]. The successful construction of a KG will bring smart functionality in various applications. With the existence of a fine-grained, high-quality KG, the search engine can understand the query such as “wife of Obama” and immediately retrieve the answer by simply recognizing “Obama” to be the text mention for the entity “Barack Obama”, and looking for the name of the entity which has the relationship “IsWifeOf” with “Barack Obama”. Besides Web search related applications, the KG can be served as a data source for building a semantic database that can be automatically used by computers, like an online Wikipedia with structured data in a strict format. It enables smart question answering systems to answer questions raised by human/computer clients in natural languages/query-like languages.

Furthermore, KGs can be served as a repository of structured knowledge which supports a large number of applications related to big data analytics. In e-commerce companies such as Walmart [5], product search and recommendation can be supported by product KGs which contain products, sellers, manufacturers, etc. The development of medicine can be benefited from KGs, where large-scale disease networks can help to enhance the medical diagnosis[1,2].

In the rest of the paper, we systematically summarize existing KG systems and analyze the state-of-the-art techniques in this field. Figure 1 illustrates the framework of a KG Based on this figure, we can get an overall picture of the field, which helps us understand the relation between each component and the entire field.

We organize the rest of the paper as follows. We identify the process of building a KG as a central task. Specifically, We review state-of-the-art techniques to extract entities from various data sources (Section 2). Then, we introduce the

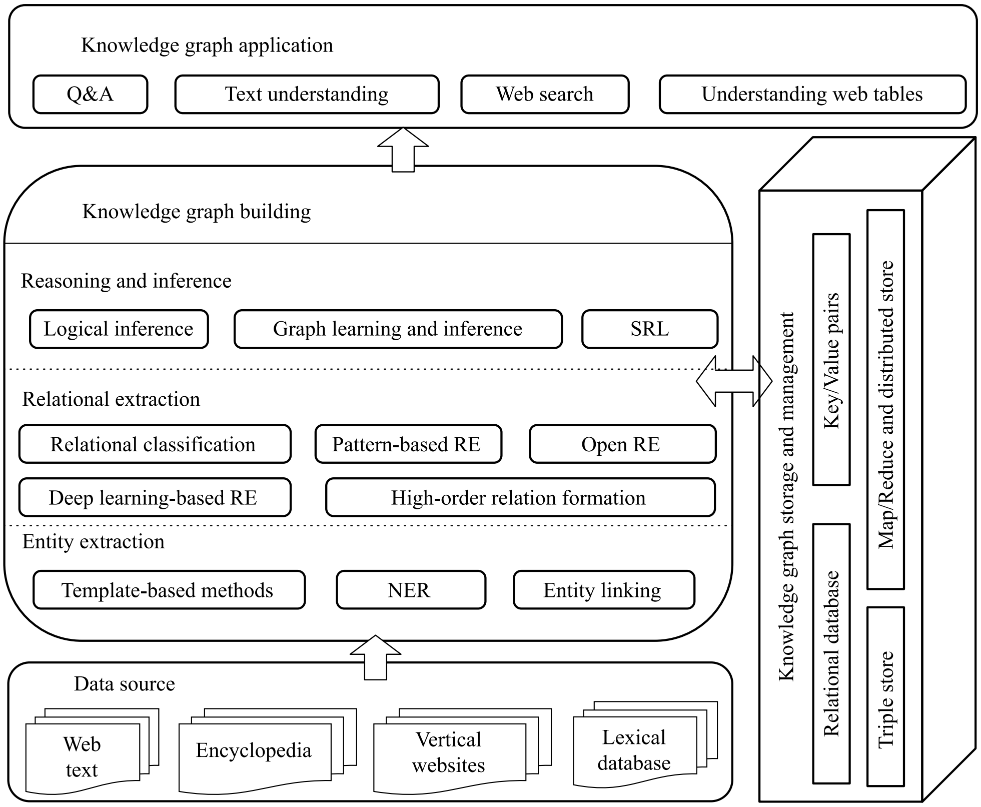

Fig. 1 The framework of KG

[1] http://dailymed.nlm.nih.gov
[2] http://www.patient.co.uk

harvest of semantic relations from texts via different approaches (Section 3). Since the extracted relations from texts tend to be incomplete and error-prone, we overview inferencing and reasoning approaches to further reduce the conflicts and improve the coverage of knowledge in Section 4. We give an overview of various graph database models and corresponding database products in both industrial and academic circles in Section 5. An overall comparison of current KGs is provided in Section 6. We discuss research issues that need to be addressed in KGs in Section 7. Finally, the conclusion is presented in Section 8.

We believe that this article provides various contributions for the KG research community, analyzing the literature on the research issues related to KGs.

2 Entity extraction

We discuss how to extract entities from various data sources in this section first. Since entities in Web pages have several surface forms, we then survey entity linking methods to link entity mentions with entities in the KG. The comparison between various methods is shown in Table 1.

Table 1 Comparison of entity mining methods

Data source	Encyclopedia	Vertical websites	Texts
Methods	Web page processing	Pattern-based method	NER
Attributes	Yes	Yes	No
Human annotation	No	No	Yes
Initial seed	No	No	No
Coverage of entities	Good	Domain specific	Depend on texts
Quality of entities	High	High	Medium

2.1 Entity extraction from semi-structured data

Wikipedia is one of the well-known data sources with high quality. Various extraction methods [6,7] have been proposed to extract contents for Wikipedia. The taxonomy and infoboxes in Wikipedia pages contain structured knowledge that can be extracted in terms of the given templates.

Besides Wikipedia, a number of high quality vertical websites provide structured/semi-structured forms for entity harvesting. As for the template-based forms, the most general solution is called wrapper induction methods [8,9]. They learn extraction wrappers from a set of manually labeled Web pages. However, it is hard to maintain and train a system when data comes from multiple web sites. In addition, there are some methods (unsupervised and semi-supervised method) [10,11] which do not need labeled training samples. They automatically produce wrappers from a collection of similar Web pages.

The template-independent methods [12,13] treat the problem of entity extraction as a classification task. Finn et al. [13] identifies the start and end tokens of a single attribute using a support vector machine. However, it fails to extract multiple entity attributes. [12] proposes 2-dimensional Conditional Random Field (2D CRF) to extract multiple attributes with two-dimensional neighborhood dependencies. Sutton et al. [14] proposes Dynamic Conditional Random Field (DCRF). They combine the advantages of both conditional random fields and the dynamic Bayesian network (DBN). The parameters of the model can be learned through an approximate inference. Wellner et al. [15] introduce a model to integrate inference and entity coreference to extract uncertainty entity. In [16], the data is modeled as hierarchical tree. A probabilistic model called Hierarchical Conditional Random Field (HCRF) is proposed to jointly optimize entity detection and attribute labeling. Given features X, and a possible label assignment, the goal of web data extraction is to compute maximum a posteriori (MAP) probability of y, then the extraction data from this assignment y^* in Eq. (1).

$$y^* = \text{argmax} p(y|X). \tag{1}$$

2.2 Entity extraction from unstructured data

Named entity recognition (NER) [17] plays a very important role in entity extraction. It focuses on identifying and classifying certain types of information elements (e.g., person, organization, location, etc.) in text, called named entities (NE). Due to the “long-tailed effect” in Web text, a large number of NEs cannot be retrieved from structured or semi-structured data. However, NER is capable of effectively populating the KG from text.

One of the most popular current NER systems is based on machine learning (ML) techniques. As for the terms/words in text, the goal of NER is to train predictive models to classify them into predefined categories (class labels). The categories mean different types of entities or the tag “none-of-the-above”, meaning that the word is not an entity of these categories. Zhou and Su [18] use a chunk tagger trained based on hidden markov model (HMM). Finkel et al. [19] propose to use conditional random field (CRF) to train a sequential NE labeler. In a sequential labeling system, the BIO (beginning, inside and outside of an entity) schema is often employed, where each term is labeled as beginning, inside or outside of a certain type of entity.

Other complex ML techniques have been applied. Liu et al. [20] combine a K-nearest neighbors (KNN) classifier with a linear CRF model to perform NER for tweets. It solves the problem of the lack of information in tweets and the unavailability of training data by a semi-supervised framework. A NER system trained on one domain is hard to be adopted to other domains due to different writing styles and lexicons. Recently, transfer learning has been studied to reduce labeling effort across different domains. Pan et al. [21] propose a transfer learning method, named Transfer Joint Embedding (TJE), for cross-domain multi class classification. In Ref. [22], Prokofyev employs external sources (e.g., DBLP, Wikipedia, etc.) to improve the effectiveness of NER.

2.3 Entity linking

An entity can be referred to multiple text mentions in documents. For example, the text “apple” can mean the concept Apple (fruit), or Apple Inc. (company) according to the context. The goal of entity linking is to link text mentions to their representation in the KG. Entity linking is an essential task to link the textual and structural information together. Specially, it helps to discover new knowledge and populate KGs.

Given a set of entity mentions M discovered in text and a set of entities E in KG, the task of entity linking to learn a function

$$f : m_i \rightarrow e_j, \text{ for } m_i \in M \text{ and } e_j \in E. \tag{2}$$

that maps a mention m_i to an entity e_j. Note that if the entity mention does not exist in the KG, NIL should be returned for the unlikable mention [23].

A number of methods have been proposed to cope with this problem. It is first regarded as a clustering task where entity mentions are clustered together so that each cluster represents one specific entity. Bagga et al. [24] use a bag-of-word (BoW) method to represent the context and apply agglomerative clustering technique based on vector space model (VSM). The work has been widely extended. In Ref. [25], Pedersen et al. employ statistically significant bigram to represent the context of a text mention. Chen and Martin [26] add a range of syntactic and semantic features to the feature vector. The context of an entity has other forms of representation as well. In Ref. [27], Lasek et al. study BoW representation, linguistic representation and structured co-occurrence representation to perform entity linking.

Websites such as Wikipedia have become a rich source and powerful tool for entity linking. Shen et al. [23] propose a framework, LINDEN, to link named entities in text with a KG. It discovers a set of candidate entities E_m for a named entity mention m. In order to rank the candidates of a given mention m, it introduces four features namely *link probability* (LP), *semantic associativity* (SA), *semantic similarity* (SS) and *global coherence* (GC). It generates a feature vector $\overrightarrow{F_m}(e) = \langle LP(e|m), SA(e), SS(e), GC(e)\rangle$ and learns a weighted vector $\overrightarrow{w}$ corresponding to $F_m(e)$. For each entity $e \in E_m$, the goal of LINDEN is to compute the maximum score, then we can find the linked entity e_{top} as follows:

$$e_{top} = \underset{e \in E_m}{\operatorname{argmax}}\, Score_m(e), \tag{3}$$

where e_{top} is the predicted mapping entity for mention m.

For semi-structured Web list data, Shen et al. [28] propose a framework LIEGE to link the entities in Web lists with the knowledge base. Based on the observation that entities mentioned in a Web list can have the same conceptual type that people have in mind, they calculate the link quality for each candidate mapping entity and then use iterative substitution algorithm to map entities in the Web list to the KG.

Graph-based methods have been adopted in entity linking. In Ref. [29], Guo et al. introduce three degree-based measures of graph connectivity to rank candidate entities by their importance then assign the most important mention to the target entity. Han et al. [30] make the assumption that entities in the same document should be semantically related, and then perform collective entity linking (CEL) in text. A referent graph is used to model the interdependence between entity linking decisions. They also propose a collective inference algorithm to link entities by exploiting the referent graph.

Probabilistic models can be also employed for this task. In Ref. [31], a three-step generative model can leverage entity knowledge for entity linking. In this model, entity knowledge is encoded in $P(e)$ (distribution of entities in document), $P(c|e)$ (distribution of possible contexts of a specific entity) and $P(s|e)$ (distribution of possible names of a specific entity). Then it links the entity e to a mention m by following function:

$$e = \underset{e}{\operatorname{argmax}} \frac{P(m, e)}{P(m)} \propto \underset{e}{\operatorname{argmax}}\, P(e)P(s|e)p(c|e). \tag{4}$$

Entity linking is associated with NER. The entities in texts must be recognized first by NER, then are linked to an existing KG. Existing approaches typically perform NER and EL using a pipelined architecture. However, NER and EL are tightly coupled. In Ref. [32], Sil and Yates propose NEREL, a joint model for NER and EL, which takes a large set of candidate mentions and a large set of candidate entity links, and ranks the candidate mention-entity pairs together to make joint prediction. Similar to NEREL, Liu et al. [33] present a graphical model to simultaneously conduct NER and named entity normalization (NEN), the task of transforming NEs in texts to their unambiguous canonical forms.

3 Relation extraction

The goal of relation extraction (RE) is to gather facts about entities with high precision and recall. In this part, we focus on the problem of extracting binary relations. A binary relation is a triple: $\langle subject, predicate, object\rangle$, in which *predicate* denotes the semantic relationship between subject and object. For example, we can get the tuple $\langle BillGates, work, Microsoft\rangle$ from the sentence "Bill Gates works at Microsoft".

In this section, we first introduce traditional supervised learning methods to extract relation instances. To avoid tedious labeling, bootstrapping systems are proposed to iteratively perform RE in large text corpora. More recently, deep learning techniques have improved the performance of RE. We introduce the advances in these areas. After that, we discuss a new framework, open relation extraction (ORE), which can identify relation tuples without any human supervision. We also address the problem of high-order RE, but it is not our focus in this paper. The high-level comparison of RE methods are shown in Table 2.

3.1 Relation classification

The RE task is generally regraded as a binary classification problem. The problem can be formalized as follows: for a sentence

$$w_1 w_2 \cdots e_1 \cdots w_i \cdots e_2 \cdots w_{n-1} w_n, \tag{5}$$

where e_1 and e_2 are two named entities and w_i denotes other words and a pre-defined relation R, the learning algorithm tries to learn a function f:

$$f(S) = \begin{cases} +1, & \text{if } e_1 \text{ and } e_2 \text{ are related by relation } R; \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where f is a binary classifier (e.g., Naives Bayes [34], Voted Perceptron [35], Log-linear [36], Maximum Entropy Model [37], etc.). S is a set of features extracted from the sentence or a structured representation of the sentence such as a labeled sequence and parse trees.

According to the nature of input of the classifier training, supervised approaches for relation extraction are further divided into feature based extraction and kernel based extraction. The process of supervised approaches is shown in Fig. 2.

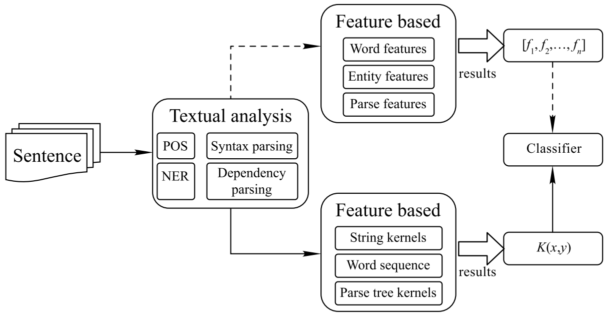

Fig. 2 The supervised approaches of relation classification

3.1.1 Feature-based extraction

Feature-based extraction in textual analysis heavily relies on NLP techniques including part-of-speech (POS) tagging, named entity recognition (NER), syntax parsing and dependency parsing. There are several sets of features used for RE shown as follows. More literatures on textual features can be found in [38].

- Word features: headwords of entities, words or bigrams in left, middle and right of the two entities, number of entities between the two entities, number of words separating the two entities, etc.
- Entity features: types of named entities (e.g., person, location) and their concatenation, mention levels (e.g., name, nominal, or pronoun), etc.
- Parse features: syntactic chunk sequence, path between the two entities in a parse tree, dependency path, etc.

Table 2 Comparison of binary relation extraction methods

Approach	Classification	Pattern-based	Deep learning	Open RE
Learning method	Supervised	Semi-supervised	Supervised	Unsupervised
Feature space	Textual features	Textual patterns	High dimensional features	Textual features
Human labeling	Yes	No	Yes	No
Initial seed	No	Yes	No	No
Predefined relation	Yes	Yes	Yes	No
Precision	High	Medium	High	Low
Recall	Low	Medium	Medium	High

3.1.2 Kernel-based extraction

Kernel-based methods are widely used in the ML community. One example is the string kernel [39], which compares text documents by the substrings they contain. In the RE task, we can extend the string kernel to complex structures such as word sequences and parse trees for RE.

In Bunescu and Mooney's work [40], a kernel is defined as the sum of the similarity of left, middle and right contexts. Each word is augmented with its POS tag, generalized POS tag, chunk tag (e.g., Noun Phrases, Verb Phrases, etc) and entity type (Person, Organization, none). For two sentences S_1 and S_2, the similarity function is:

$$\begin{aligned} sim(S_1, S_2) = K(left_1, left_2) + K(mid_1, mid_2) +\\ K(right_1, right_2), \end{aligned} \tag{7}$$

where $K(\cdot,\cdot)$ is the kernel function for two sub-sentences.

Kernel methods can be applied in complex tree structures. Zelenko et al. [41] introduce kernels defined over shallow parse trees and devise it in conjunction with SVM and voted perceptron (VP) algorithm. Shallow parsing only identifies its key elements rather than the full interpretation of a sentence. A node in the parsing tree is defined as a set of attributes $\{a_1,a_2,\ldots\}$, and each node necessarily has attributes with names "Type" and "Role". Given two related examples P_1 and P_2, the kernel function is defined as follows:

$$\begin{aligned} &K(P_1, P_2)\\ &= \begin{cases} 0, \text{ if } t(P_1.p, P_2.p) = 0;\\ k(P_1.p, P_2.p) + K_c(P_1.c, P_2.c), \quad \text{otherwise}, \end{cases} \end{aligned} \tag{8}$$

where $P_i.p$ and $P_i.c$ represent the parent and child nodes of P_i respectively. $k(P_1.p, P_2.p) \in \{0, 1\}$ is a similarity function between nodes $P_i.p$ and $P_2.p$. $K_c(P_1.c, P_2.c)$ is a similarity function of the subtrees.

Culotta and Sorensen [42] extend previous tree kernels with richer structured features to estimate the similarity between the dependency trees. In Ref. [43], Bunescu and Mooney design a generalization of subsequence kernels which uses three types of subsequence patterns for RE.

In conclusion, supervised methods have been widely studied. It performs well and results in high precision, but it is formidable to extend to new relation types due to lack of training data. Also, textual NLP analysis like POS tagging, syntax parsing and dependency parsing is a necessity and this step is prone to error.

3.2 Pattern-based relation extraction

Supervised learning methods require a large number of human-labeled training data, which is quite tedious and not feasible for large-scale Web RE. A major trend for extracting relation instances of interest is to use textual patterns. Textual pattern analysis for RE dates back to Hearst patterns [44]. Hearst patterns are hand-crafted, hyponymic, lexico-Syntactic patterns between two or more noun phrases. In the pattern

$$NP_0 \text{ such as } \{NP_1, NP_2, \ldots, (\text{and}|\text{or})\}\ NP_n, \tag{9}$$

we can infer the rule

$$\text{for all } NP_i, 1 \leqslant i \leqslant n, \text{hyponym}(NP_i, NP_0), \tag{10}$$

where NP_i is a noun phrase identified by a POS tagger. Although Hearst patterns are effective and enjoy relatively high precision, patterns for a given relation are defined by hand. Also it suffers from low recall since manually defining all patterns are not possible and there are no unified patterns for some relations.

Pattern-based RE systems employ a bootstrapping approach to iteratively discover relation instances and patterns. It uses seeds to directly learn to populate a relation. Basic procedure for extracting relation instances for relation R is shown as follows: first, we gather a set of seed pairs that have relation R. Then, we scan the corpus to find sentences with these pairs. Patterns can be generated by looking at the context between or around the pair. By pattern matching techniques, we can generate more relations instances. The process is performed iteratively until certain convergence criteria is reached, e.g., no new relation instances can be found. Since only a few seed tuples or a few high-precision patterns are needed instead of a large training set, the system can run in a semi-supervised or unsupervised way (or weakly supervised learning).

In dual iterative pattern expansion (DIPRE) [45], Brin exploits the duality between sets of patterns and relations to grow the target relation starting from a small sample (seed). The system uses seed examples and web crawlers to find all documents containing the pair. It generalizes matched sentences in the documents into wild card expressions like ⟨*longest-common-suffix of prefix strings, .*?, middle, .*?, longest-common-prefix of suffix strings*⟩.

The project Snowball [46], similar to DIPRE, also initializes its bootstrapping system by a set of seeds, then it finds segments in context where the two target entities occur close to each other to generates patterns. A Snowball pattern is a 5-tuple $\langle left, tag1, middle, tag2, right\rangle$ where *tag*1 and *tag*2 are NE tags and $left$, *middle* and *right* are weighted feature vectors. The weights for every term are defined as the normalized frequency of the word appeared in a given position. The similarity distance between two 5-tuples $t_p = \langle l_p, t_1, m_p, t_2, r_p \rangle$ and $t_s = \langle l_s, t'_1, m_s, t'_2, r_s \rangle$ is defined in Eq. (11):

$$Match(t_p, t_s) = \begin{cases} l_p \cdot l_s + m_p \cdot m_s + r_p \cdot r_s, & \text{if the tags match;} \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

It groups tuples into classes through a simple single-pass clustering algorithm. Finally it adopts those generalized patterns to populate new candidate tuples. The main components of snowball are shown in Fig. 3.

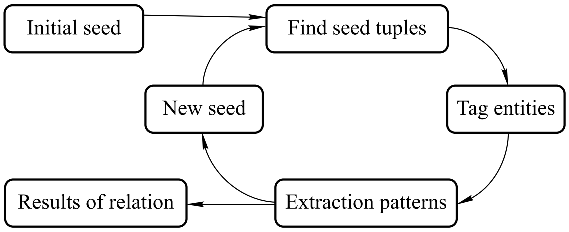

Fig. 3 The framework of snowball

The KnowItAll system [47] takes the advantage of search engines to perform autonomous, domain-independent extraction. It is a bootstrapping system which consists of two main components: *Extractor* and *Assessor*. It starts the bootstrapping system by a set of hand written patterns based on a NP (Noun Phrase) chunker, which are more general than Hearst patterns, to generate extraction rules and "discriminator" phrases. Such typical patterns include

$$NP_1 \text{ is the } NP_2 \text{ of } NP_3, \tag{12}$$

$$\text{the } NP_1 \text{ of } NP_2 \text{ is } NP_3, \tag{13}$$

where NP_i represents a noun phrase. It utilizes the *Extractor* component to extract the candidate facts from Web pages returned from search engines. It automatically estimates the plausibility of those candidate facts using pointwise mutual information (PMI) statistics. The *Assessor* component treats those statistics as features of a classifier to predict whether these extraction results are correct or not.

In Ref. [48], Nakashole et al. present a scalable system PROSPERA to achieve knowledge harvesting with high precision, high recall and scalability. There are three phases in the PROSPERA system: 1) *Pattern gathering*, mainly aimed at gathering relation patterns from sentences. 2) *Pattern analysis*, which analyzes the basic patterns from previous step and to generate the *lifted patterns* which characterize the original patterns' important features. It replaces the basic patterns' words with their POS tags. Then, it calculates the *support* and *confidence* of a lifted pattern. 3) *Reasoning*, to improve precision, it uses a series of manually specified constraints to prune false positive facts.

In conclusion, relation bootstrapping systems require fewer human supervision and continue to enjoy popularity in information extraction. However, the biggest challenge is to achieve both high precision and high recall at the same time. Strict patterns with strong rules and constraints ensure high precision while neglecting recall. The opposite approach yields high recall and low precision. To construct KGs automatically, achieving both is necessary, which is the recent research focus.

3.3 Open relation extraction

Traditional RE systems using supervised or semi-supervised methods take a relation name and a handful of seeds or a set of training data as inputs. The relation types are predefined by designers or domain experts, which is formidable to scale to Web RE over massive, heterogeneous corpora. Banko et al. [49] introduces a system TextRunner. It makes a single pass over corpus and extracts a large number of relation instances without any human inputs. It uses a self-supervised learner, a classifier that can label candidate extractions as trustworthy or not. It scans the corpus and computes a probability of a relation tuple being correct using the probabilistic model [50].

Although open RE systems can accumulate a massive body of knowledge automatically, due to the lack of human supervision, two types of errors are frequently occur: incoherent extractions and uninformative extractions [51]. Incoherent extractions are extracted relation phrases that have no meaningful interpretation. Uninformative extractions are extractions that omit critical information. To cope with this problem, syntactic and lexical constraints are added to yield richer and more informative relations. Etzioni et al. [52] propose the second generation of open RE, where patterns are treated as features to identify arguments given a sentence and a relation phrase pair.

Open RE aims at finding all relation instances from Web data instead of the limited predefined relations. It provides a framework to mine all relation facts from the Web. However, this type of approaches unavoidably have several disadvantages. All verbs between nouns or noun phrases have a probability to be identified as a relation, while it is not the case in the real world. The flexibility results make it difficult to perform inference in KGs. Also, it still suffers from the low precision problem, which makes it less appealing in real-world applications.

3.4 Word embedding-based relation extraction

Word embedding is a collection of techniques for natural language processing where words or phrases are mapped to vectors in a low-dimensional space. Techniques to generate these maps include deep learning, matrix factorization and topic modeling, etc.

In open relation extraction (ORE), the relations are detected from an explicit space, for instance clustering [49]. Instead of clustering, [53] uses matrix factorization techniques to map entities into an implicit space. This work proposes non-negative matrix factorization (NNMF) and boolean matrix factorization (BMF) to discover the relations between entities after mapping entities into a low-dimensional space.

With rapid development of deep learning techniques, complex and deep models on text mining and natural language processing can be employed to extract knowledge more accurately. The fundamental concept of such deep learning techniques is to compute distributed representations of words, also known as word embeddings [54]. Neural-based representation learning methods encode KGs with low-dimensional vector representations of both entities and relations, which can be further used to extract unknown related facts [55]. Jason Weston et al. propose a weakly supervised approach for relation extraction from free text which is trained to jointly use information from the text and from existing knowledge [56]. Relation mentions, entities and relations in this method are all embedded into a common low-dimensional vector space. A ranking-based embedding framework is used to train the model. For a triple (h, r, t), the model learns the plausibility of the triple by generalizing from the KG. Mo Yu et al. propose a new method to jointly learn features embeddings and a tensor-based classifier for relation extraction task [57]. In the approach, the number of parameters is dramatically reduced since features are only represented as low-dimensional embeddings.

3.5 High-order relation formation

Most of the relations considered in this survey are binary relations. There exist higher-order relations as well. An n-ary relation can be defined as an n-tuple $\langle e_1, e_2, \ldots, e_n\rangle$ where all e_i are entities w.r.t the certain relation. For instance, from a sentence "Satya Nadella is the CEO of Microsoft.", we can extract an instance $(SatyaNadella, CEO, Microsoft)$ of a ternary relation $T = (people, job, company)$. Although it is not straightforward to adapt to higher-order relations using methods we present in this survey, we can factor higher-order relations into a set of binary relations. As a result, the previously mentioned relations can be factorized into $hasPosition(SatyaNadella, CEO)$ and $worksIn(SatyaNadella, Microsoft)$.

McDonald et al. [58] construct higher-order relations by combining binary relations. Binary relations are first extracted via a classifier. Entities which have relations are connected into a entity graph, then higher-order relations can be extracted by finding maximal cliques in the graph.

4 Knowledge graph reasoning and inference

Because of the data extracted from the web contains noise, facts (i.e., entities and relations) tend to be incomplete and error-prone. As a result, there is a genuine need to improve the coverage and to reduce the conflicts in automatically constructed KGs.

In this section, we review three major classes of approaches that are able to efficiently populate the KG as follows:

- Rule learning using logical inference;
- Graph-based inference and learning algorithms;
- Entity and relation embedding based inference;
- Statistical relational learning (SRL) [59] approaches, such as Markov Logic Networks (MLN).

4.1 Logical inference

There exist rules between relations in KGs. Rules are based on literals, which are statements that have placeholders for entities, e.g., $bornInCitycity(x, y)$, $cityInCountry(y, z)$ and $cityInCountry(y, z)$. Rules can be expressed via Horn clauses [60]. Horn clauses are disjunction of literals containing at most one positive literal. For example, The rule among these literals can be expressed as

$$\begin{aligned} \neg bornInCity(x, y) \lor \neg cityInCountry(y, z); \\ \lor bornInCountry(x, z), \end{aligned} \tag{14}$$

which is logically equivalent to the following rule:

$$\begin{aligned} bornInCity(x, y) \land cityInCountry(y, z) \rightarrow; \\ bornInCountry(x, z). \end{aligned} \tag{15}$$

Rules have certain properties in different KGs. Rules can reveal inconsistencies between relation instances already in KGs. It is simple to prove that relation instances

$bornInCity(Bob, Shanghai)$ and $cityInCountry(Shanghai, China)$ have a conflict with $bornInCountry(Bob, US)$. Rules can be soft or hard in different settings. Soft rules can be violated in some degree while hard rules must always be hold. For soft rules, they can be weighted to handle uncertain or probabilistic data [61].

The first-order learning system emerged even before KGs in the AI community, e.g., GOLEM [62] and FOIL [63]. It constructs first-order Horn clauses (rules) from data examples. The system takes positive and negative examples as input and then learns a set of Horn clauses that fit the data well. The learned rules can be used to infer new relation instances from ones that already exist in KGs. These rules can also be served as consistency constraints to prune false relation instances.

NELL [64] employs a variant of FOIL named N-FOIL [65] as the rule learner. It learns probabilistic Horn clauses in a "separate-and-conquer" manner. To improve its scalability, it assumes the consequent predicate is functional. For instance, each *Person* is born at most in one *City*. It also uses a "relational pathfinding" [66] strategy to produce general rules. The conditional probability $P(conclusion|preconditions)$ of each N-FOIL rule is estimated using a multinomial distribution with a Dirichlet prior.

SOFIE [61] is the first system that integrates logical consistency reasoning with information extraction (IE). It casts known facts, hypotheses for new facts, word-to-entity mappings, patterns and constraints into logical clauses. It assigns weights to clauses that are derived from statistical evidence in the data. It aims at finding true clauses such that a maximum number of constraints is satisfied and casts the problem into a weighted maximum satisfiability problem (Weighted MAX-SAT). It can be solved by customizing MAX-SAT solvers.

4.2 Graph learning and inference

To improve the converge, various algorithms have been proposed to directly perform inference on the graph to generate new relation instances.

Several graph algorithms related to random walks can be used in inferencing. One popular measure is random walk with restart (RWR) [67]. Lao et al. propose the path ranking algorithm (PRA) [68] for relational retrieval. It learns to rank graph nodes y relative to a query node x in the graph. Each node x together with any other node y is treated as a training query. For a relation R, if the pair x and y is known to satisfy $R(x, y)$, then it is labeled as a positive example; otherwise, it is a negative example. Lao et al. [65] apply this approach to perform learning and inference tasks in a large-scale knowledge base using "Data-Driven Path Finding". For a large-scale KG, sampling methods are required to generate a subset of relation tuples to perform scalable learning and inference, various sampling methods can be performed such as particle filtering [69] and low-variance sampling (LVS) [70]. Gardner et al. [71] further use latent syntactic cues for inferencing, which outperforms the previous PRA approach.

Related to PRA, Wang et al. propose using personalized PageRank (ProPR) [72] for graph inferencing. It is an extension to "stochastic logic programs (SLPs)" [73]. The randomized traversal of G is defined by a probabilistic choice at each node. Each edge is associated with a feature vector with their respective weights and each node has an edge pointed to the start node inspired by RWR. Parameters are learned via stochastic gradient descent (SGD), which can be easily adapted to parallel learning tasks [74].

4.3 Entity and relation embedding based inference

The goal of knowledge graph completion is to perform link prediction between entities. However, techniques for the traditional link prediction is not capable for knowledge graph completion because: 1) vertices in KGs are entities associated with different types and attributes; 2) edges in KGs are relations of different types; 3) for KG completion, we need to determine if there is a relation between two entites or not, as well as the relation type if it exists. There are some existing works to study the KG completion based on the entity and relation embedding techniques.

Neural tensor network (NTN) [75] is one of the most popular models, which provides a more powerful way to present relational information than a standard neural network layer. It models with a bilinear tensor layer that directly relates the two entity vectors across multiple dimensions. For a triple (h, r, t), NTN defines a score function for graph embedding as follows:

$$f_r(h, t) = u_r^{\perp} g(h^{\perp} M_r t + M_{r,1} h + M_{r,2} t + b_r), \tag{16}$$

where u_r is a relation-specific linear layer, $g()$ is the *tanh* operation, M_r is a 3-way tensor, and $M_{1,r}$ and $M_{2,r}$ are weight matrices. Recursive neural tensor network (RNTN) [76] is a recursive version of NTN. RNTN takes phrases of any length as inputs, and represents a phrase through word vectors and a parse tree and then compute vectors for higher nodes in the tree using the same tensor-based composition function. However, the high complexities of both NTN and RNTN may prevent them from efficient applying on large-scale KG embeddings.

TransE learns vector embeddings for both entities and relations [77]. It translates entity h into entity t via using relation r when (h, r, t) holds. This indicates that t should be the nearest neighbor of $h + r$ if (h, r, t) holds. Hence TransE defines the score function as follows:

$$f_r(h, t) = ||h + r - t||_2^2. \tag{17}$$

The score is low if (h, r, t) holds, and high otherwise. However, TransE is not capable of modeling 1-to-N, N-to-1 and N-to-N relations.

To model 1-to-N, N-to-1 and N-to-N relations, TransH [78] is proposed to embed a KG in a continuous vector space. It allows an entity to have distinct representations when involved in different relations. For a triple (h, r, t), the entity embeddings h and t are first projected to the hyperplane of w_r as the normal vector, denoted as $h_\perp$ and $t_\perp$. Hence, the score function is defined as

$$f_r(h, t) = ||h_\perp + r - t_\perp||_2^2. \tag{18}$$

However, both TransE and TransH assume embeddings of entities and relations being in the same low-dimensional space. In practice, an entity may have multiple aspects, and various relations exist in different aspects.

TransR is proposed to model entities and relations in different spaces in terms of different aspects of entities. It builds entity and relation embeddings in separate entity space and relation spaces [79]. For each relation r, a projection matrix $M_r \in \mathbf{R}^{k\times d}$ may project entities from entity space to relation space. With the mapping matrix, the projected vectors of entities are defined as $h_r = hM_r$ and $t_r = tM_r$. Hence the score function can be defined as

$$f_r(h, t) = ||h_r + r - t_r||_2^2. \tag{19}$$

In this way, TransR represents entities and relations in different semantic space bridged by relation specific matrices. Furthermore, the dimensions of entity and relation embeddings are not necessarily identical.

Besides TransE, TransH and TransR, there are also many other methods following the KG embedding associated with the different score functions, such as unstructured model (UM) [80], structured embedding (SE) [81], single layer model (SLM) [75], and latent factor model (LFM) [82].

In summary, the study of deep learning for relation extraction is gaining popularity in academic circles. Methods based on word embedding provide a framework to represent entity and relation instances in a high dimensional space so that the hidden correlation of entities and relations can be found in the high dimensional space. These approaches have high precision in knowledge extraction and improve the recall by discovering relations that cannot be captured by traditional methods.

4.4 Statistical relational learning

Statistical relational learning (SRL) [59] involves models that can represent both uncertainty and relational structure at the same time. It provides a general framework to perform learning and inferencing tasks in a machine learning approach. Among these models, Markov logic network (MLN) proves to be most versatile in relation learning [83]. Without losing the completeness, we review other models that are frequently appeared in the literature as well.

4.4.1 Markov logic network

First-order rules put a set of hard constraints on data in KGs. Even if there exists one relation instance that violates the rules, it has zero probability. However, it is often the case that conflicts and uncertainty appear in unreliable data source, e.g., Web data. To soften these constraints, the MLN [84] has been proposed. An MLN is a simple representation which combines probabilistic graphical models and first-order logic. It has a weight attached to each formula (rule), indicating how strong the formula is. Thus instances with violation of constraints are less probable, but not completely impossible.

In the Markov Logic framework, an MLN can be viewed as a template for constructing Markov networks. Each literal can be interpreted as a random variable. The literals in clauses are represented as probabilistic dependencies between random variables. Random variables and their dependencies form a Markov random field (MRF) [85]. The inference task in KGs can be performed in MRF corresponding to MLN. Due to the NP-hardness of inferencing in MRF, various algorithms can be employed for joint inference such as Markov chain monte carlo (MCMC) methods [86] (e.g., Gibbs sampling [87]), belief propagation [88], etc.

In the literature, StatSnowball [89] employs general discriminative MLNs such as logistic regression (LR) and conditional random fields (CRF) [90] to be the statistical models. It uses l_1-regularized feature selection method to discover patterns and identify relations. The weights of generated patterns can be automatically inferred from the MLN model.

4.4.2 Probabilistic models with constraints

Unlike MLNs, Chang et al. [91] propose constraints conditional models (CCM) that separate the probabilistic and the declarative aspects of the model. It allows the probabilistic portion of the model to represent an arbitrary conditional distribution without modeling the probability of input variables. On the other hand, prior knowledge can be encoded in the form of constraints. It has been applied to tasks such as IE and semantic role labeling (SRL). Carlson et al. [92] integrate constraints into semi-supervised learning algorithms. The key idea is coupling multiple functions to constrain learning problems and iteratively training classifiers in a self-supervised manner. Any candidates violating mutual exclusion and type checking constraints should be filtered.

5 Knowledge graph storage and management

Due to the wide spread of graph data models and graph algorithms, as well as the different application scenarios for graph data, a wide range of database models, systems and query languages have been defined for graph databases. Rather than having a unified model for all graph databases, each graph database, or each type of graph databases is designed for a specific kind of tasks. Also, no standard data definition and manipulation language has been introduced for graph databases. Most databases implement their own APIs for data management instead of the standard SQL.

In this section, we give an overview of various graph database models and corresponding database products in both industrial and academic communities.

5.1 Relational database

Although traditional RDBMS is formidable to be employed for big graph management, it is extremely fast in querying, inserting, deleting, and updating information in database tables. Because there is no golden standard for graph data storage, relational databases are still increasingly used. Among these, a number of graph databases are implemented on the top of existing RDBMS.

G-store [93] is a storage prototype for large vertex-labeled graphs. It is equipped with a built-in query engine that has the functionality of various graph algorithms, such as depth-first traversal, reachability testing, shortest path search and shortest path tree search. It is on the top of PostgreSQL [94], an object-relational database system.

Filament[3] is a project for storing and managing graph data structures and provides a graph persistence framework and associated toolkits. It has the functionality of supporting SQL through JDBC for querying graph data. It uses a fluent traversal model that can navigate the stored graph in large chunks.

5.2 Key/Value pairs

The key/value store proves to be vital in the world of NoSQL database systems. The application of key/value stores greatly improve the scalability of the graph databases. It is easy to deploy key/value storage systems in distributed clusters as a scale-out architecture. Another advantage is that the key/value model simplifies the traditional relational model and allows more flexibility in data types.

Trinity [95] is a general purpose distributed graph database system developed by Microsoft Research. It provides in-memory key/value store over a cluster of machines. It has the features of both low-latency online query processing and high-throughput offline analytics on large scale graphs with billions of nodes.

VertexDB[4] is a high performance graph database using key/value disk storage. In VertexDB, nodes are folders of key/value pairs and keys are stored in lexical ordering for fast retrieval. Currently, its implementation is built based on TokyoCabinet [96] (a key/value disk store), libevent and Yajl in C.

CouchDB [97] is open-source key-value store of the Apache CouchDB project. It supports a sophisticated replication mechanism on massive graph data. Mondal et al. [98] implement a distributed graph data management based on CouchDB, which supports graph partition over large dynamic graphs.

CloudGraph[5] is a .NET graph database that uses graphs and key/value pairs to store graph data. It is disk- and memory-based system with the features of hot backup, pluggable storage and resource-balancing. It is fully transactional and has a traversal framework with graph query language (GQL).

5.3 Triple store

The unified framework for representing information in the Web is the resource description framework (RDF)[6]. The store of RDF is based on W3C RDF and SPARQL standards. In RDF, each directed and labeled edge is represented by a triple: ⟨*subject, predicate, object*⟩. *predicate* is the label for the relation from the subject vertex to the object vertex. An

[3] http://filament.sourceforge.net

[4] http://www.dekorte.com/projects/opensource/vertexdb

[5] http://www.cloudgraph.com

[6] http://www.w3.org/RDF

RDF graph can be viewed as a (potentially huge) set of such triples, and each component of a triple is a RDF term which is consist of three types: internationalized resource identifier (IRI), literal and blank nodes. In a triple, a subject is an IRI or a blank node; a predicate must be an IRI and an object is an IRI, a literal or a blank node. A number of graph databases focus on storing RDF triples for storage. These techniques can be utilized for knowledge data storage and management.

AllegroGraph [99] is a close source, high-performance, persistent RDF graph database used to store triples. It uses SPARQL, the W3C standard RDF query language, to query RDF data. It also has the reasoning function through RDFS++ Reasoning as the reasoner and Prolog as the alternative. AllegroGraph has various applications as well, e.g., geospatial and temporal reasoning, social network analysis, etc.

Neo4j [100] is a open source, embedded, disk-based graph database implemented in Java. It is highly scalable and fully supports the properties of atomicity, consistency, isolation and durability (ACID). As a network-oriented database, it can extend to several clusters to process billions of nodes in parallel. Neo4j has been applied in mission-critical applications.

DEX [101] is also a high-performance, scalable graph DBMS implemented in Java and C++. In DEX, A DEX graph is a labeled directed attributed multigraph (LDAM), making it suitable for storage of complex graph structures. The transactions in DEX support aCiD, meaning full consistency and durability support with partial isolation and atomicity.

Sones[7] is an object-orientated graph DBMS in a distributed environment. It is suitable for managing a large amount of highly connected semi-structured data. The system has its own query language based on SQL and other complex queries on the graph as well.

HyperGraphDB [102] is a general purpose, open source data storage based on "directed hypergraphs". It is an embedded, transactional graph database for large scale knowledge storage. It provides a universal data model for applications such as bioinformatics and natural language processing.

5.4 Map/Reduce and distributed storage

The Map/Reduce paradigm can be employed to process large graph efficiently and effectively in a parallel computing manner. Rather than using triple store, this class of graph databases focus on partitioning large number of nodes to different machines. Each machine only needs to do a relatively small size of computation using Map/Reduce.

One Map/Reduce platform that is worth mentioning is Hadoop[8]. Hadoop allows for distributed processing of massive data sets across computer clusters. It offers reliable and scalable computation for offline data processing but is not readily suitable for graphs. Inspired by the Map/Reduce framework, Malewicz et al. [103] propose Pregel. In this system, programs are treated as a sequence of iterations called supersteps. In each superstep, each vertex can receive messages sent in the previous iteration, compute a certain function in parallel and send to other vertices. There are other graph database implementations on top of Pregel, including Phoebus[9] and Giraph[10], in order to take advantage of the Map/Reduce framework.

Besides the Hadoop's Map/Reduce framework, there are other graph databases using distributed storage as well. InfiniteGraph [104] is a distributed-oriented system that combines the strengths of persisting and traversing complex relationships requiring multiple hops.

In conclusion, there are various graph databases nowadays available or under development, most of which are application-driven. Since there is no standard graph data model, database system or query language, the choice of graph databases is based on its applications.

6 Knowledge graph system overview

In previous sections, we have described KGs and the techniques behind them. In this section, we first give summaries of influential KGs. Then we do a detailed comparison between these projects.

6.1 Summaries of knowledge graphs

- CYC
  CYC [105], started in 1984, was to codify millions of pieces of knowledge that composes human common sense in machine-usable form. The CYC knowledge base contains over one million human-defined assertions, rules and common sense ideas, formulated in the language CYCL, which is based on predicate calculus. At the present time, the CYC KB contains over five hundred thousand terms, including about seventeen thousand types of relations, and about seven million assertions re-

[7] http://github.com/sones/sones

[8] http://hadoop.apache.org

[9] https://github.com/xslogic/phoebus

[10] https://github.com/apache/giraph

lating these terms. New assertions are continually added to the KB through a combination of automated and manual means. The CYC inference engine applies logical rules to the knowledge base and derived answers from a knowledge base.

- ConceptNet
  ConceptNet [106] is created by open mind common sense (OMCS) project. The goal is to build and utilize a large commonsense knowledge base from the contributions of many thousands of people across the Web. Since 1999, it has accumulated more than a million English facts from over 15 000 contributors and knowledge bases in other languages. A semantic network builds from this corpus is called ConceptNet, which is used for natural language processing, understanding and commonsense reasoning.
- WordNet
  WordNet [107] was initially conceived as a lexical database for machine translation. Currently, WordNet is used as a semantic network and as an ontology. It contains 117 000 synsets, which are groups of synonyms corresponding to a concept. These synsets connect with each other through several semantic relations. WordNet has also been extended to a multi-lingual version, MultiWordNet [108].
- HowNet
  HowNet [109] is a commonsense knowledge base which unveils inter-conceptual relations and inter-attribute relations of concepts in the form of Chinese and their English equivalents. There is an important concept in HowNet: sememe, which is the smallest basic semantic unit that cannot be reduced further, such as “human being”. Currently, there are over 800 sememes in HowNet. These sememes are extracted through the examination of about 6 000 Chinese characters.
- FrameNet
  The FrameNet [110] project is to build a lexical database for English. It contains more than 10 000 word senses, most of which have annotated examples that show the meaning and usage of the entry. The project also provides more than 170 000 manually annotated sentences, which are served as a training dataset for NLP tasks, such as sentiment analysis, machine translation, information extraction, etc.
- KnowItAll
  KnowItAll [47] is an information extraction system which performs unsupervised, domain-independent extraction tasks from the Web. It uses the search engine to perform extraction from massive Web data, which outperforms previous systems in the tasks of pattern learning, subclass extraction and list extraction.
- NELL
  Never-ending language learner (NELL) [64] is a knowledge base implemented by the ReadTheWeb Project. It keeps populating a growing knowledge base of structured facts. Given an initial ontology containing 123 categories and 55 relations, it is able to extract 242 453 beliefs with an estimated precision of 74% in 67 days. So far, NELL has accumulated over 50 million candidate beliefs by reading the Web.
- TextRunner
  TextRunner [49] is an information extraction system. In contrast to traditional information extraction, it does not need human-labeled training data or relation seeds. It can read any text from any domain on the Web, extracts meaningful information and stores in a knowledge base for querying. Currently, TextRunner has successfully extracted over 500 million assertions from 100 million Web pages.
- Probase
  Probase [111] is a probabilistic knowledge base consisting of about 2.7 million concepts. The concepts are extracted from a corpus of 1.68 billion Web pages. To model inconsistent and uncertain data, it uses probabilistic models to build a probabilistic taxonomy. The goal of Probase is to understand human communication in text using common-sense knowledge or general knowledge.
- Freebase
  Freebase [112] is a graph-shaped database of structured general human knowledge. It is a stable, practical platform for collecting knowledge by crowd sourcing. The current data in stored Freebase consists of millions of concepts (topics) and tens of millions of relationships between those topics.
- DBpedia
  DBpedia [7] is a multi-lingual knowledge base in which the structured contents are extracted from Wikipedia. The structural knowledge in DBpedia is accumulated using crowd-sourced techniques. DBpedia contains 24.9 million things in 119 languages, including 4.0 million in English.
- Kylin/KOG
  Kylin/KOG [113–115] is an autonomous computer sys-

tem that builds a rich ontology by combining Wikipedia infoboxes and WordNet. Each infobox is treated as a class and it is simultaneously mapped to WordNet nodes. It also maps attributes between related classes. Kylin/KOG is an application of Markov logic network (MLN) and other learning techniques.

- YAGO/YAGO2
  YAGO/YAGO2 [6,116,117] is a huge semantic knowledge base in which the knowledge is extracted from Wikipedia and other sources. In 2014, it contains more than 10 million entities (e.g., persons, cites, organizations, etc.) and more than 120 million facts about these entities.
- Google knowledge graph
  Google knowledge graph (GKG) [18] is a knowledge base used by Google to add semantic search functionality to its search engine. Google's search engine provides structural information about the topic inferred from user's query using GKG. The KG has compiled more than 3.5 billion facts over 500 million objects or entities.
- Satori
  Satori is a KG developed by Microsoft to support Bing search. The knowledge in Satori comes from websites such as social network websites (e.g., LinkedIn), information providers (e.g., Vitals), etc.
- Facebook graph search
  Facebook graph search [119] provides semantic search service by Facebook. It combines data acquired from over one billion users and external data to provide user-specific search results. Users can search for pages, people, places, check-ins, etc. using natural languages.
- EntityCube / Renlifang
  Renlifang [89] is a entity relationship search engine in Chinese. The entities and relationships are mined from Chinese web pages using StatSnowball [89]. The English version of Renlifang is EntityCube.
- OpenCalais
  OpenCalais[11] automatically creates rich semantic metadata for the Web content. Using NLP, ML and other methods, it analyzes the document and finds entities in it. Besides classic entity identification, it returns the facts and events hidden within the text as well.
- Sogou Zhilifang
  Zhilifang[12] is the first Chinese KG used in search engine that powered by Sogou[13]. Currently, the number of entities is over 100 million and the number of relations between entities is over one million.
- Baidu Zhixin
  Baidu Zhixin[14] is the KG that provides semantic search functionality for Baidu search engine[15]. It is an information integration search system centralized in different industries. It uses data mining techniques to search for answers based on data on the Internet.

6.2 Comparison

We compare and evaluate these KGs in an unified framework. Furthermore, we hope to discover potential supplement and enhancement to existing works. We first compare KGs involved in this article item by item on the basis of several important features characterizing the systems. Table 3 summarizes the comparison results.

The first column (Project) denotes the KG names. The language of KGs are English by default. For exceptional cases, we use **C** as abbreviation for Chinese and **M** for multi-languages. The second column (Affiliation) denotes the development teams or organizations. They are mostly large Internet companies, or notable research teams. The third column (Year) represents the development time of the systems, which are arranged in ascending order.

The fourth column (KGs content) denotes the contents inside KG systems. We can compare the data scale clearly. The fifth column (Data source) describes where the data is from.

The sixth column (Data object) defines what kind of knowledge is to be acquired, which mainly includes fact, ontology and lexical. We use ontology to represent KGs that contain ontological information, fact to represent KGs that contain factual information and lexical to represent KGs that contain lexical information. It is obvious that a KG can have multiple data types. For example, YAGO combines an ontology derived from WordNet and facts from Wikipedia.

The seventh column is KG storage. It is used to describe the data storage format, including RDF, XML, OWL, XML, JSON, etc. The eighth and ninth columns are Entity Extraction and Relation Extraction, respectively. They directly describe the knowledge harvesting methods of KGs.

[11] http://www.opencalais.com
[12] http://roll.sohu.com/20121122/n358347929.shtml
[13] http://www.sogou.com
[14] http://tech.163.com/13/1023/08/9BRUO0UV000915BF.html
[15] http://www.baidu.com

Table 3 Comparison of KGs

Project	Affiliation	Year	KGs content	Data source
CYC	Cycorp	1984	120K+ concepts	Human-defined
WordNet(M)	PU	1985	117K synsets	Expert-authored
FrameNet(M)	UC Berkeley	1997	10 000 word senses and 170 000 annotated sentences	Human-defined
ConceptNet	MIT	1999	1.6M assertions	OMCS
HowNet	Keenage	2000	800 sememes	Modern Chinese dictionary
KnowItAll	UW	2005	54 753 facts	Web pages
TextRunner	UW	2007	500M assertions	117M+ Web pages
Freebase	Metaweb	2007	22M+ entities	Crowdsourced
DBpedia	DBpedia	2007	320 classes, 0.7M Wiki types, 3.6M entities, 247 M triples	Wikipedia
YAGO/YAGO2	MPII	2007	10M+ entities, 120M+ facts	Wikipedia, WordNet
Kylin/KOG	UW	2008	Classified documents and sentences	Wikipedia infoboxes and WordNet
OpenCalais	Calais	2008	Semantic metadata	Web Pages
Satori	MS	2009	400M entities	Bing search streams
NELL	CMU	2010	123 categories and 55 relations	500M Web pages
Probase	MSRA	2011	2.7M concepts	1.68B Web pages
EntityCube/Renlifang(C)	MSRA	2011	Entity-Relation Graph	Billions of public Web pages
Google Knowledge Graph(M)	Google	2012	570M+ objects, 18B facts	CIA World Factbook, Freebase, Wikipedia
Sogou Zhilifang(C)	Sogou	2012	100M+ entities, 1M+ relations	Web pages
Baidu Zhixin(C)	Baidu	2012	N/A	Web pages
Facebook Graph Search	Facebook	2013	User's network of friends, Bing's search engine	700TB post and comment data from Facebook blog

Project	Data object	KG storage	Entity extraction	Relation extraction
CYC	Ontology, Facts	CycL	Human-defined	Human-defined
WordNet(M)	Ontology, Facts	OWL/RDF	Human-defined	Human-defined
FrameNet(M)	Ontology, Facts	XML/OWL	Human-defined	Human-defined
ConceptNet	700K Facts	JSON	Template-based	Infobox of Wiki
HowNet	6,000 Chinese characters	Named sememes	Template-based	Pattern-based
KnowItAll	XML	Pattern/Wrapper/Rule	Feature-based classification	
TextRunner	Facts	RDF	Pattern/Wrapper	Feature-based
Freebase	Ontology, Facts	RDF	Crowdsourced	Crowdsourced
DBpedia	Ontology, Facts	RDF	Template-based	Infobox of Wiki
YAGO/YAGO2	Ontology, Facts	RDF	Template-based/Rule	Pattern-based/Rule
Kylin/KOG	Ontology, Facts	RDF	Pattern/Wrapper/Rule	CRF-based classification
OpenCalais	N/A	OWL/RDF	Template-based	Pattern-based
Satori	Ontology, Facts	RDF	Search log/Feature-based	Pattern-based
NELL	Ontology, Facts	TSV File	Pattern-based	Kernel/Feature-based
Probase	Is-A pair	RDF	Hearst-pattern/Search log	Pattern-based
EntityCube/Renlifang(C)	Facts	RDF	Feature-based	Pattern-based
Google knowledge graph(M)	Ontology, Facts	RDF Triple	Crowdsourced	Crowdsourced
Sogou Zhilifang(C)	Ontology, Facts	N/A	Search log/Vertical website	Pattern-based/Feature-based
Baidu Zhixin(C)	Ontology, Facts	N/A	Pattern/Vertical website/Search log	Feature-based
Facebook graph search	Facts	GraphML	Template-based	Pattern-based

In Table 3, we can observe that: 1) the sizes of knowledge graphs have become larger and larger; 2) recently, almost all of big Internet companies in the world provide the services based on the knowledge graph techniques. These changes benefit from the explosion in the data volume and the progress in the techniques.

7 Discussion

Although the construction of large-scale KGs has been extensively studied in both academic (e.g., YAGO [116], DBPedia [7]) and industry circles (e.g., Google Knowledge Graph, Microsoft's Satori), there remains a lots of research issues that still need to be addressed. We discuss some of the issues as follows.

7.1 Web data cleansing

Unlike data sources such as Wikipedia that contain high-quality data but with limited size, the Web provides an unprecedented amount of data for knowledge extraction. However, one challenge of constructing KGs from the Web is to generate high-quality knowledge from noisy Web data. The cleansing of Web data can greatly improve the data accuracy. For example, leveraging probabilistic approaches to select values for the attributes that are the most likely correct.

7.2 Temporal and spatial knowledge extraction

The correctness of some facts in KGs is preserved only under some temporal and spatial constraints. For instance, the president of the United States may change over time. Freebase allows users to add beginning and end dates to attributes of entities. YAGO2 [6] integrates knowledge from WordNet, Wikipedia and GeoNames to annotate facts with temporal and spatial information. In future research, it is vital to study how to extract these facts from the Web automatically.

7.3 Knowledge graph construction for different languages

The techniques of KG construction are strongly related to the processing of natural language itself. Methods for KG construction in English cannot be directly applied to KG in other languages. Here, we take Chinese as an example. Currently, most projects introduced in this paper focus on constructing English KGs while several challenges remain for Chinese KG construction [120]. Data sources in Chinese are still limited due to the lack of knowledge repositories (e.g., Freebase), low quality of Wikipedia in Chinese and the difficulty in integrating knowledge from heterogeneous data sources. Establishing classes and their semantic relations are formidable in Chinese. Extraction rules and patterns that work well for the English language do not work in Chinese. Thus, much research work should be devoted to mapping entities to their corresponding classes and building the entire taxonomy. Expressions in Chinese language are quite flexible, which results in increased difficulty in building large-scale Chinese entity and relation extraction systems. The advancement in this field will enrich current Chinese KGs.

7.4 Knowledge graph construction in specific domains

A majority of existing approaches for knowledge extraction from the Web are based on Web data redundancy. For example, a lot of relation extraction systems mine relational facts by discovering patterns that are frequently expressed in Web text, such as Snowball [46], StatSnowball [89], etc. These projects focus on harvesting a vast amount of general knowledge. However, when it comes to a specific domain, the Web documents related to the domain are relatively sparse, which makes it difficult to use pattern-based methods. Therefore, it is important to research how to extract entities and relations for a specific domain.

7.5 Knowledge fusion from existing knowledge graphs

In this paper, we survey different fact extraction methods and different KGs. To construct a KG system with larger scale, we can combine facts extracted from different methods and facts from existing KGs together. In this way, the problem of knowledge fusion calls for future research. To specify, facts from different data sources and systems should have different levels of "correctness" that need to be estimated. Knowledge extracted from plain text has lower probability to be correct than that directly derived from Freebase. After the assignment of "correctness" of these facts, a general framework should be developed to fuse these facts and integrate them into a unified system.

8 Conclusion

In this survey, we have provided an analysis of state-of-the-art techniques in the fields of KGs. In more details, we have discussed the main challenges in constructing, managing and storing KGs and approaches to deal with these issues.

The building of a KG is the central topic in this survey. To study this topic in detail, we have described methods for mining entities from different data sources and relation extraction methods in different paradigms. We have also presented several techniques to improve the quality of data in KGs, including logical reasoning and inference. We analyze the storage and management of graph data in KGs and survey existing KG systems from different aspects.

To conclude, in recent years, KGs have emerged as one of the most important systems for knowledge representation, organization and understanding. A large number of research issues, applications and products related to KGs have been proposed. Nevertheless, there are still many challenges and opportunities in the field of KGs.

Acknowledgements This work has been supported by the National Key Research and Development Program of China (2016YFB1000905), the National Natural Science Foundation of China (Grant Nos. U1401256, 61402177, 61402180) and the Natural Science Foundation of Shanghai (14ZR1412600). This work was also supported by CCF-Tecent Research Program of China (AGR20150114). The author would also like to thank Key Disciplines of Software Engineering of Shanghai Second Polytechnic University (XXKZD1301).

References

1. Frost A. Introduction to Knowledge Base Systems. Macmillan Publishing Company, Inc., 1986
2. Hua W, Song Y Q, Wang H X, Zhou X F. Identifying users' topical tasks in Web search. In: Proceedings of the 6th ACM International Conference on Web Search and Data Mining. 2013, 93–102
3. Song Y Q, Wang H X, Wang Z Y, Li H S, Chen W Z. Short text conceptualization using a probabilistic knowledgebase. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. 2011, 2330–2336
4. Wang J J, Wang H X, Wang Z Y, Zhu K Q. Understanding tables on the Web. In: Proceedings of the International Conference on Conceptual Modeling. 2012, 141–155
5. Deshpande O, Lamba D, Tourn S, Subramaniam S, Rajaraman A, Harinarayan V, Doan A. Building, maintaining, and using knowledge bases: a report from the trenches. In: Proceedings of ACM Special Interest Group on Management of Data. 2013, 1209–1220
6. Hoffart J, Suchanek F, Berberich K, Weikum G. YAGO2: a spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence, 2013, 194: 28–61
7. Lehmann J, Isele R, Jakob M, Jentzsch A, Kontokostas D, Mendes P, Hellmann S, Morsey M, Kleef P, Auer S, Bizer C. DBpedia — a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web Journal, 2015, 6(2): 167–195
8. Kushmerick N. Wrapper induction: efficiency and expressiveness. Artificial Intelligence, 2000, 118: 15–68
9. Muslea I, Minton S, Knoblock C. Hierarchical wrapper induction for semistructured information sources. Autonomous Agents and Multi-Agent Systems, 2001, 4(1–2): 93–114
10. Buttler D, Liu L, Pu C. A fully automated object extraction system for the World Wide Web. In: Proceedings of ACM International Conference on Distributed Computing Systems. 2001, 361–370
11. Wang R, Cohen W. Language-independent set expansion of named entities using the Web. In: Proceedings of the 7th IEEE International Conference on Data Mining. 2007, 342–350
12. Nie Z, Wen J R, Zhang B, Ma W Y. 2D conditional random fields for Web information extraction. In: Proceedings of the 22nd International Conference on Machine Learning. 2005, 1044–1051
13. Finn A, Kushmerick N. Multi-level boundary classification for information extraction. In: Proceedings of the 15th European Conference on Machine Learning. 2004, 111–122
14. Sutton C, Rohanimanesh K, Mccallum A. Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. In: Proceedings of the 21st International Conference on Machine Learning. 2004, 693–723
15. Wellner B, Mccallum A, Peng F, Hay M. An integrated, conditional model of information extraction and coreference with application to citation matching. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence. 2004, 593–601
16. Zhu J, Nie Z, Wen J R. Simultaneous record detection and attribute labeling in Web data extraction. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2006, 494–503
17. Marrero M, Urbano J, Sánchez-Cuadrado S, Morato J, Berbís J. Named entity recognition: fallacies, challenges and opportunities. Computer Standards & Interfaces, 2013, 35(5): 482–489
18. Zhou G D, Su J. Named entity recognition using an hmm-based chunk tagger. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, 2002, 473–480
19. Finkel J, Grenager T, Manning C. Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2005, 363–370
20. Liu X H, Zhang S D, Wei F R, Zhou M. Recognizing named entities in tweets. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2011, 359–367
21. Pan S J, Toh Z, Su J. Transfer joint embedding for cross-domain named entity recognition. ACM Transactions on Information Systems, 2013, 31(2): 7
22. Prokofyev R, Demartini G, Cudré-Mauroux P. Effective named entity recognition for idiosyncratic Web collections. In: Proceedings of the 23rd International Conference on World Wide Web. 2014, 397–408
23. Shen W, Wang J Y, Luo P, Wang M. Linden: linking named entities with knowledge base via semantic knowledge. In: Proceedings of the 21st International Conference on World Wide Web. 2012, 449–458
24. Bagga A, Baldwin B. Entity-based cross-document coreferencing using the vector space model. In: Proceedings of the 17th International Conference on Computational Linguistics. 1998, 79–85.
25. Pedersen T, Purandare A, Kulkarni A. Name discrimination by clustering similar contexts. In: Proceedings of Computational Linguistics and Intelligent Text Processing. 2005, 226–237
26. Chen Y, Martin J. Towards robust unsupervised personal name disambiguation. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2007, 190–198
27. Lasek I, Vojtás P. Various approaches to text representation for named entity disambiguation In: Proceedings of International Conference on Information Integration and Web-based Applications and Services. 2013, 242–259
28. Shen W, Wang J Y, Luo P, Wang M. Liege: link entities in Web lists with knowledge base. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2012, 1424–1432
29. Guo Y H, Che W X, Liu T, Li S. A graph-based method for entity linking. In: Proceedings of the 5th International Joint Conference on Natural Language Processing. 2011, 1010–1018
30. Han X P, Sun L, Zhao J. Collective entity linking in Web text: a graph-based method. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2011, 765–774
31. Han X P, Sun L. A generative entity-mention model for linking entities with knowledge base. In: Proceedings of Annual Meeting on Association for Computational Linguistics, 2011, 945–954

32. Sil A, Yates A. Re-ranking for joint named-entity recognition and inking. In: Proceedings of the 6th Workshop on Ph.D. Students in Information and Knowledge Management. 2013, 2369–2374

33. Liu X H, Zhou M, Zhou X F, Fu Z, Wei F. Joint inference of named entity recognition and normalization for tweets. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2012, 526–535

34. Russell S, Norvig P. Artificial intelligence: a modern approach. New Jersey: Prentice-Hall, Egnlewood Cliffs, 1995, 25

35. Collins M, Duffy N. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2002, 263–270

36. Knoke D, Burke P. Log-linear Models. New York: SAGE Publications. 1980, 20

37. Ratnaparkhi A. A maximum entropy model for part-of-speech tagging. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 1996, 133–142

38. Kambhatla N. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In: Proceedings of the ACL on Interactive Poster and Demonstration Sessions. 2004, 20

39. Lodhi H, Saunders C, Shawe-Taylor J, Cristianini N, Watkins C. Text classification using string kernels. Journal of Machine Learning Research, 2002, 2(3): 419–444

40. Bunescu R, Mooney R. A shortest path dependency kernel for relation extraction. In: Proceedings of Conference on Human Language Technology and Empirical Methods in Natural Language Processing. 2005, 724–731

41. Zelenko D, Aone C, Richardella A. Kernel methods for relation extraction. Journal of Machine Learning Research, 2003, 3(3): 1083–1106

42. Culotta A, Sorensen J. Dependency tree kernels for relation extraction. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2004, 423–429

43. Bunescu R, Mooney R. Subsequence kernels for relation extraction. Advances in Neural Information Processing Systems. 2005, 171–178

44. Hearst M. Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics. 1992, 539–545

45. Brin S. Extracting patterns and relations from the World Wide Web. In: Proceedings of WebDB Workshop at the 6th International Conference on Extending Database Technology. 1998, 172–183

46. Agichtein E, Gravano L. Snowball: extracting relations from large plain-text collections. In: Proceedings of the 5th ACM International Conference on Digital Libraries. 2000, 85–94

47. Etzioni O, Cafarella M, Downey D, Popescu A, Shaked T, Soderland S, Weld D, Yates A. Unsupervised named-entity extraction from the Web: an experimental study. Artificial Intelligence, 2005, 165(1): 91–134

48. Nakashole N, Theobald M, Weikum G. Scalable knowledge harvesting with high precision and high recall. In: Proceedings of the 4th ACM International Conference on Web Search and Data Mining. 2011, 227–236

49. Banko M, Cafarella M, Soderland S, Broadhead M, Etzioni O. Open information extraction from the Web. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence. 2007, 2670–2676

50. Downey D, Etzioni O, Soderland S. A probabilistic model of redundancy in information extraction. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence. 2005, 1034–1041

51. Fader A, Soderland S, Etzioni O. Identifying relations for open information extraction. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 2011, 1535–1545

52. Etzioni O, Fader A, Christensen J, Soderland S. Open information extraction: the second generation. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. 2011, 3–10

53. Cergani E, Miettinen P. Discovering relations using matrix factorization methods. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management. 2013, 1549–1552

54. Bian J, Gao B, Liu TY. Knowledge-powered deep learning for word embedding. In: Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2014, 132–148

55. Lin Y K, Liu Z Y, Luan H B, Sun M S, Rao S W, Liu S. Modeling relation paths for representation learning of knowledge bases. 2015, arXiv:1506.00379

56. Weston J, Bordes A, Yakhnenko O, Usunier N. Connecting language and knowledge bases with embedding models for relation extraction. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013, 1366–1371

57. Yu M, Gormley M, Dredze M. Combining word embeddings and feature embeddings for fine-grained relation extraction. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2015, 1374–1379

58. McDonald R, Pereira F, Kulick S, Winters R, Jin Y, White P. Simple algorithms for complex relation extraction with applications to biomedical IE. In: Proceedings of Annual Meeting on Association for Computational Linguistics. 2005, 491–498

59. Getoor L, Taskar B. Introduction to statistical relational learning. Journal of the Royal Statistical Society: Series A (Statistics in Society), 2010, 173(4): 934–935

60. Ueda K. Guarded horn clauses. In: Proceedings of the Conference on Logic Programming. 1985, 168–179

61. Suchanek F, Sozio M,Weikum G. Sofie: a self-organizing framework for information extraction. In: Proceedings of the 18th International Conference on World Wide Web. 2009, 631–640

62. Muggleton S, Feng C. Efficient induction of logic programs. Inductive Logic Programming, 1992, 38: 281–298

63. Quinlan J, Cameron-Jones R. Foil: a midterm report. In: Proceedings of the European Conference on Machine Learning. 1993, 3–20

64. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka Jr. E R, Michell T M. Toward an architecture for never-ending language learning. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence. 2010, 5: 3

65. Lao N, Mitchell T, Cohen W. Random walk inference and learning in a large scale knowledge base. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2011, 529–539

66. Richards B, Mooney R. Learning relations by pathfinding. In: Pro-

ceedings of the 10th National Conference on Artificial Intelligence. 1992, 50–55

67. Tong H, Faloutsos C, Pan J. Random walk with restart: fast solutions and applications. Knowledge and Information Systems, 2008, 14(3): 327–346

68. Lao N, Cohen W. Relational retrieval using a combination of path-constrained random walks. Machine Learning, 2010, 81(1): 53–67

69. Kotecha J, Djuric P. Gaussian particle filtering. In: Proceedings of IEEE Transactions on Signal Processing. 2003, 51(10): 2592–2601

70. Thrun S, Burgard W, Fox D. Probabilistic robotics (intelligent robotics and autonomous agents series). Intelligent Robotics and Autonomous Agents, 2002, 45(3): 52–57

71. Gardner M, Talukdar P, Kisiel B, Mitchell T. Improving learning and inference in a large knowledge-base using latent syntactic cues. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013, 833–838

72. Wang W, Mazaitis K, Lao N, Mitchell T, Cohen W. Efficient inference and learning in a large knowledge base: reasoning with extracted information using a locally groundable first-order probabilistic logic. 2014, arXiv:1404.3301

73. Cussens J. Parameter estimation in stochastic logic programs. Machine Learning, 2001, 44(3): 245–271

74. Zinkevich M, Weimer M, Smola A, Li L. Parallelized stochastic gradient descent. In: Proceedings of Advances in Neural Information Processing Systems. 2010, 2595–2603

75. Socher R, Chen D, Manning C, Ng A. Reasoning with neural tensor networks for knowledge base completion. In: Proceedings of Advances in Neural Information Processing Systems. 2013, 926–934

76. Socher R, Perelygin A, Wu J, Chuang J, Manning C, Ng A, Potts C. Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013, 1631: 1642–1653

77. Bordes A, Usunier N, García-Durán A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. Advances in Neural Information Processing Systems, 2013, 2787–2795

78. Wang Z, Zhang J W, Feng J L, Chen Z. Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence. 2014, 1112–1119

79. Lin Y K, Liu Z, Sun M S, Liu Y, Zhu X. Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence. 2015, 2181–2187

80. Bordes A, Glorot X, Weston J, Bengio Y. Joint learning of words and meaning representations for open-text semantic parsing. In: Proceedings of the 15th International Conference on Artificial Intelligence and Statistics. 2012, 127–135

81. Bordes A, Weston J, Collobert R, Bengio Y. Learning structured embeddings of knowledge bases. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence. 2011

82. Sutskever I, Salakhutdinov R, Tenenbaum J. Modelling relational data using bayesian clustered tensor factorization. Advances in Neural Information Processing Systems, 2009, 1821–1828

83. Weikum G, Theobald M. From information to knowledge: harvesting entities and relationships from Web sources. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. 2010, 65–76

84. Richardson M, Domingos P. Markov logic networks. Machine learning, 2006, 62(1–2): 107–136

85. Kindermann R, Snell J. Markov Random Fields and Their Applications. Providence: American Mathematical Society. 1980

86. Poon H, Domingos P, Sumner M. A general method for reducing the complexity of relational inference and its application to mcmc. In: Proceedings of the 23rd National Conference on Artificial Intelligence. 2008, 1075–1080

87. Resnik P, Hardisty E. Gibbs Sampling for the Uninitiated. Technical Report, DTIC Document. 2010

88. Duchi J, Tarlow D, Elidan G, Koller D. Using combinatorial optimization within max-product belief propagation. In: Proceedings of Advances in Neural Information Processing Systems. 2006, 369–376

89. Zhu J, Nie Z, Liu X, Zhang B, Wen J R. Statsnowball: a statistical approach to extracting entity relationships. In: Proceedings of the 18th International Conference on World Wide Web. 2009, 101–110

90. Lafferty J, McCallum A, Pereira F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning. 2001, 282–289

91. Chang M, Ratinov L, Rizzolo N, Roth D. Learning and inference with constraints In: Proceedings of the 23rd Conference AAAI on Artificial Intelligence. 2008, 1513–1518

92. Carlson A, Betteridge J,Wang R, Hruschka Jr. E, Mitchell T. Coupled semi-supervised learning for information extraction. In: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining. 2010, 101–110

93. Das S, Agrawal D, Abbadi A. G-store: a scalable data store for transactional multi key access in the cloud. In: Proceedings of the 1st ACM Symposium on Cloud Computing. 2010, 163–174

94. Momjian B. PostgreSQL: introduction and concepts. New York: Addison-Wesley, 2001

95. Shao B, Wang H X, Li YT. Trinity: A distributed graph engine on a memory cloud. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. 2013, 505–516

96. Hirabayashi M. Tokyo Cabinet: A Modern Implementation of DBM. http://1978th.net/tokyocabinet/, 2010

97. Anderson J, Lehnardt J, Slater N. CouchDB: the Definitive Guide. Sebastopol: O'Reilly Media Inc., 2010

98. Mondal J, Deshpande A. Managing large dynamic graphs efficiently. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. 2012, 145–156

99. Aasman J. Allegro graph: RDF triple database. Technical Report. 2006

100. Miller J. Graph database applications and concepts with Neo4j. In: Proceedings of the Southern Association for Information Systems Conference. 2013, 141–147

101. Martinez-Bazan N, Gomez-Villamor S, Escale-Claveras F. Dex: a high-performance graph database management system. In: Proceedings of IEEE International Conference on Data Engineering Workshops. 2011, 124–127

102. Iordanov B. Hypergraphdb: a generalized graph database. In: Proceedings of the 11th International Conference on Web-Age Information Management Workshops. 2010, 25–36

103. Malewicz G, Austern M, Bik A, Dehnert J, Horn I, Leiser N, Czajkowski G. Pregel: a system for large-scale graph processing. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. 2010, 135–146

104. Lans R. Infinitegraph: extending business, social and government intelligence with graph analytics. The Analysis, 2010

105. Matuszek C, Cabral J, Witbrock M, DeOliveira J. An introduction to the syntax and content of Cyc. In: Proceedings of AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering. 2006, 44–49

106. Liu H, Singh P. Conceptnet a practical commonsense reasoning toolkit. BT Technology Journal, 2004, 22(4): 211–226

107. Miller G. Wordnet: a lexical database for English. Communications of ACM, 1995, 38(11): 39–41

108. Magnini B, Strapparava C, Ciravegna F, Pianta E. A project for the constraction of an italian lexical knowledge base in the framework of wordnet. In: Proceedings of International Workshop on the "Future of the Dictionary". 1994

109. Zhang Z D, Dong Q. Hownet — a hybrid language and knowledge resource. In: Proceedings of Natural Language Processing and Knowledge Engineering. 2003, 820–824

110. Baker C, Fillmore C, Lowe J. The berkeley framenet project. In: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics. 1998, 86–90

111. Wu W, Li H, Wang H X, Zhu K Q. Probase: a probabilistic taxonomy for text understanding. In: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. 2012, 481–492

112. Bollacker K, Cook R, Tufts P. Freebase: a shared database of structured general human knowledge. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence. 2007, 1962–1963

113. Weld D, Hoffmann R, Wu F. Using wikipedia to bootstrap open information extraction. ACM SIGMOD Record, 2008, 37(4): 62–68

114. Wu F, Weld D. Autonomously semantifying wikipedia. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management. 2007, 41–50

115. Wu F, Weld D. Automatically refining the wikipedia infobox ontology. In: Proceedings of the 17th International World Wide Web Conference. 2008, 635–644

116. Suchanek F, Kasneci G, Weikum G. Yago: a core of semantic knowledge. In: Proceedings of the 16th International World Wide Web Conference. 2007, 697–706

117. Biega J, Kuzey E, Suchanek F. Inside YAGO2s: a transparent information extraction architecture. In: Proceedings of the 22nd International World Wide Web Conference. 2013, 325–328

118. Singhal A. Introducing the knowledge graph: things, not strings. Official Google Blog, 2012

119. Sengupta S. Facebook unveils a new search tool. New York Times, 2013

120. Wang C Y, Gao M, He X F, Zhang R. Challenges in Chinese knowledge graph construction. In: Proceedings of the 31st IEEE International Conference on Data Engineering Workshops. 2015, 59–61

Jihong Yan is a PhD candidate in Institute for Data Science and Engineering, East China Normal University, China. She is an associate professor at the Institute for Computer and Information Engineering, Shanghai Second Polytechnic University, China. Her research interests include Web data management and mining.

Chengyu Wang is a PhD candidate in Institute for Data Science and Engineering, East China Normal University, China. His research interests include Web data mining and information extraction. He is currently working on constructing Chinese knowledge graphs from Web data sources.

Wenliang Cheng is a graduate student of Institute for Data Science and Engineering, East China Normal University, China. His research interests include knowledge graph and natural language processing.

Ming Gao is an associate professor at the Institute for Data Science and Engineering, East China Normal University, China. He received his PhD in computer science from Fudan University, China. Prior to joining ECNU, Ming Gao worked with Living Analytics Research Centre (LARC), Singapore Management University, Singapore. His current research interests include distributed data management, user profiling and social mining, data stream management and mining, and uncertain data management.

Aoying Zhou is a professor of computer science at East China Normal University, China, where he is heading the Institute of Massive Computing. He is the winner of the National Science Fund for Distinguished Young Scholars supported by NSFC and the professorship appointment under Changjiang Scholars Program of Ministry of Education.