A Meta Path based Method for Entity Set Expansion in Knowledge Graph

Yuyan Zheng, Chuan Shi, *Member, IEEE,* Xiaohuan Cao, Xiaoli Li, *Senior Member, IEEE,* and Bin Wu, *Member, IEEE*

Abstract—Entity Set Expansion (ESE) is the problem that expands a small set of seed entities into a more complete set, entities of which have common traits. As a popular data mining task, ESE has been widely used in many applications, such as dictionary construction, query suggestion and new brand identification. Existing ESE methods mainly utilize text and Web information. That is, the intrinsic relation among entities is inferred from their occurrences in text or Web. With the surge of knowledge graph in recent years, it is possible to extend entities according to their occurrences in knowledge graph. In this paper, we consider the knowledge graph as a heterogeneous information network (HIN) that contains different types of objects and links, and propose a novel method, called MP_ESE, to extend entities in the HIN. The MP_ESE employs meta paths, a relation sequence connecting entities, in HIN to capture the implicit common traits of seed entities. In addition, an automatic meta path generation method, called SMPG, has been designed to exploit the potential relations among entities. Heuristic learning and PU learning methods are employed to learn the weights of extracted meta paths. With these generated and weighted meta paths, the MP_ESE can effectively extend entities. Comprehensive experiments on real datasets show the effectiveness and efficiency of MP_ESE.

Index Terms—Heterogeneous information network, Entity set expansion, Knowledge graph, Meta path, PU learning.

1 Introduction

Entity Set Expansion (ESE) refers to the problem of expanding a small set of a few seed entities into a more complete set, entities of which have some implicit common features and belong to a particular class. The particular class can be any collection of entities that are conceptually consistent and seeds are the instances of entities in the set. For example, given a few seeds like "China","America" and "Russia" of country class, ESE will leverage data sources (e.g., text or Web information) to obtain other country instances, such as Japan and Korea. ESE has been used in many applications, e.g., dictionary construction [1], word sense disambiguation [2], query refinement [3], query suggestion [4] and new brand [5].

Numerous methods have been proposed for ESE and these methods can be summarized into three categories, based on the text corpus, based on Web environment and others [6], [7], [8], [9], [10]. The first two categories mainly utilize distribution information or context pattern of seeds to expand entities. For instance, Wang and Cohen [7] propose a novel approach that can be applied to semi-structured documents written in any markup language and in any human language. Others leverage knowledge graph, which is a popular tool to store and retrieve fact information with graph structure, such as Wikipedia [1] and Yago [11], as auxiliary for the performance improvement of ESE in text or Web. For example, Qi et al. [12] use Wikipedia semantic knowledge to choose better seeds for ESE. However, there is few work that only utilizes knowledge graph to study the problem of ESE. It is necessary to employ knowledge graph as individual data source for ESE based on the following reasons. (1) Traditional methods based on text or Web information need complex natural language process which affects the performances to a large extent. However, it needs not additional pre-processing for knowledge base. (2) Knowledge base includes rich objects and semantic relations, which may be useful for ESE.

In this paper, we first study the entity set expansion with knowledge graph. Since knowledge graph is usually constituted by $< Subject, Predicate, Object >$ tuples, we can consider it as a heterogeneous information network (HIN) [13], which contains different types of objects and relations. Based on this HIN, we design a novel *M*eta *P*ath based *E*ntity *S*et *E*xpansion approach (called MP_ESE). Specifically, the MP_ESE employs the meta path [14], a relation sequence connecting entities, to capture the implicit common features of seed entities. Because the number of meta paths between two entities in knowledge graph is huge, we can not enumerate them manually. Therefore, we design an automatic meta path generation method, called SMPG, to exploit the potential relations among entities. Although meta paths have been discovered, the importance of different meta path is different. Heuristic learning and PU learning methods (Learning from Positive and Unlabeled Examples) are employed to effectively learn the weights of meta paths. With the help of weighted meta paths, MP_ESE can automatically extend entity set. Particularly, based on the Yago knowledge graph, we generate six different types of entity set expansion tasks. On almost all tasks, the proposed method outperforms other baselines. In addition, our

- *Y.Y. Zheng, C. Shi, X.H. Cao and B. Wu are with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing, China. E-mail: zyy0716_source@163.com, devil_baba@126.com, {shichuan,wubin}@bupt.edu.cn.*
- *X.L. Li is with the Institute for Infocomm Research, A*STAR, Singapore. E-mail: xlli@i2r.a-star.edu.sg.*

Manuscript received February 28, 2017.

1. https://en.wikipedia.org/

method can provide more insights by mining the subtle semantic meaning among entities.

The original work has been published in [15]. However, this paper substantially extends the previous work in the following aspects. First, we employ a new weight learning method, called PU learning, to automatically determine the weights of extracted meta paths. The additional experiments validate its effectiveness. Second, in order to extensively evaluate the proposed method from different perspectives, we add two baselines, two expansion tasks and four experiments which are of great significance for the ESE problem. Third, we add the related works about knowledge graph, HIN and meta path, as well as the intensive introductions about the preliminary and proposed method.

The remainder of the paper is organized as follows. We first summarize the related work in Section 2 and then describe the preliminary in Section 3. We illustrate the proposed method in Section 4. Section 5 presents the experiment details, as well as the experiment results. Finally, the paper concludes in Section 6.

2 Related Work

In this section, we review some related work in terms of knowledge graph, HIN, meta path and entity set expansion.

Knowledge graph [16] has been introduced to optimize the search results by Google in 2012. Since then, there have been a wide variety of descriptions and definitions about knowledge graph [17]. Knowledge graph is a knowledge base system with semantic properties and is often constructed from semi-structured knowledge, such as Wikipedia, or harvested from the web with a combination of statistical and linguistic methods [18]. In order to infer and add missing knowledge to the knowledge graph as well as to increase its utility, various existing refinement methods have been proposed [18]. Considerable researches into the tasks of utilizing knowledge graph have also been carried out in recent years, especially in the Q&A (natural language question and answer) system [19], [20], [21]. For example, Zou et al. [21] propose a systematic framework to answer natural language questions over RDF repository (RDF Q&A) from a graph data-driven perspective.

Heterogeneous information network(HIN), consisting of different types of objects and relations, has two important characteristics: complex structure and rich semantics. It provides a new paradigm to manage networked data. Many data mining tasks have been studied in HIN, which include similarity measure [14], [22], [23], clustering [24], [25], classification [26], [27], link prediction [28], [29], ranking [30], [31], recommendation [32], [33] and so on [13]. In order to better study the knowledge graph, we can model knowledge graph by different ways. For example, we can create a statistical model for knowledge graph data and use a tensor to represent it. Nickel et al. [34] review relational machine learning methods for knowledge graph. In this paper, we model knowledge graph as a HIN, as it has strong capability to better model complex structure and rich semantics that are needed for representing knowledge graph.

As a unique characteristic and effective semantic capture tool in HIN, meta path has been adopted in many data mining tasks, such as similarity measure, clustering, classification and so on. Meta path is a sequence of relation types connecting object types. Since different types of objects and links coexist in HIN and they contain different semantic meaning, some data mining tasks based on different meta paths may obtain different results. Sun et al. [14] first propose the meta path concept and study the meta path-based top-k similarity search in HIN. Afterwards, Shi et al. [23] propose a general framework for relevance measure based on meta path in HIN, which can measure the relatedness of objects with the same or different types in a uniform framework. Besides the fundamental similarity measure problem, Sun et al. [35] use meta path to control clustering with distinct semantics and integrate meta-path selection with user-guided clustering to cluster objects in HIN. Yu et al. [36] introduce meta-path-based latent features to represent the connectivity between users and items along different types of paths, and then define recommendation models at both global and personalized levels. Recently, Shi et al. [37] propose to study HIN relevance from a probabilistic perspective and design a generative model to derive a novel path-based relevance measure called PReP. Although meta path has shown its powerful ability in many mining tasks, it has not yet been adopted to solve the ESE problem so far. In this paper, we firstly employ meta path to capture the rich semantic meaning between entities so as to address the ESE problem in knowledge graph.

In recent years, there has been a significant amount of work on ESE. It has received considerable attention from both research [7], [8], [38], [39] and industry community (e.g., Google Sets). According to the difference of data sources utilized by ESE, these methods are based on text, Web environment and others.

Specifically, for those text data source based ESE methods, there is a basic hypothesis that words with similar meanings tend to appear in similar contexts [40]. They utilize the distribution information of the surrounding words of entities or context feature selection mechanism to expand certain class [6], [9], [10], [41]. For those Web environment based ESE methods, proper patterns of seeds are extracted and then these patterns are used to extract new candidate entities. This kind of methods can also be used in text data source. For example, Shi et al. [42] propose a probabilistic Co-Bootstrapping method so as to resolve the expansion boundary problem and the semantic drift problem. In addition, Krishnan et al. [43] exploit semantic information from Wikipedia and propose a method called Select-Link-Rank to generate diversified query expansions. In order to leverage the inherent relationship between entities and attributes, Zhang et al. [44] propose a joint model for entity set expansion and attribute extraction. Bing et al. [45] develop a new framework to achieve the goal of Wikipedia entity expansion and attribute extraction. Recently, some researchers begin to take advantage of external semantic information to improve performance of entity set expansion for text or Web data source. For instance, Qi et al. [12], [46] introduce the semantic knowledge by leveraging Wikipedia and reduce the seed ambiguity. Sadamitsu et al. [47] use topic information to alleviate semantic drift. Jindal et al. [48] specify some negative examples to confine the expansion class.

More recently, HIN and knowledge graph have also

been applied for entity search or entity query. Yu et al. [49] propose a meta-path-based ranking model ensemble to represent semantic meaning for entity query. QBEES [50], [51] is designed for entity similarity search based on aspects of the entities. Fetahu et al. [52] propose a two-fold entity retrieval approach including offline preprocessing and optimized retrieval model. Kahng et al. [53] propose a probabilistic entity retrieval model for RDF graphs using paths in the graph. Chen et al. [54] design a system for entity exploration and debugging. They do not employ the knowledge graph and the HIN method at the same time. They assume that an entity could be described with entities, types or relations that are directly linked to. But they ignore those entities or relations being indirectly linked to, which can also describe an entity to some extent, even contain important information. Even though Yu et al. [49] adopt the meta path, however, the meta paths in their methods need to be provided by domain expert users, which is obviously time-consuming and costly, especially for large-scale dataset. We not only consider the indirect relationships between entities and employ the meta path to capture the rich semantic meaning between entities, but also design a novel method to automatically discover the important meta paths between entities in knowledge graph.

3 PRELIMINARY

In this section, we describe some key concepts and present some preliminary knowledge in this paper.

Definition 1. ***Knowledge graph (KG)*** *[16], [17]. A knowledge graph is defined as an RDF graph. An RDF graph consists of a set of RDF triples in the form* $< Subject, Predicate, Object >$*, where* $Subject$*,* $Predicate$ *and* $Object$ *are all RDF terms, an RDF term is either a URI (uniform resource identifiers), a blank node, or a literal.*

In Fig. 1, $< Steven\ Spielberg, directed, War\ Horse\ (film) >$ is an example of RDF triples. Yago [11], DBpedia [55] and Freebase [56] are prime examples of KG. The types of entities or relations in KG are often organized as concept hierarchy structure, which describes the sub-class relationship among entity types or relations. Fig. 1(b) is a snapshot of Yago concept hierarchy structure and we can see that actor is sub-class of person shown by the dashed line in Fig. 1(b). All these types share a common root called Thing.

In Yago dataset, there are altogether 7 abstract taxonomies under the root Thing. They are person, organization, building, yagoGeoEntity, artifact, abstraction and physical entity, respectively, two of which are showed in Fig. 1(b). Each class belongs to a certain branch of the taxonomy. The Yago concept hierarchy has four levels in all. The root level is Thing, the 7 abstract taxonomies belong to the second level, the third level is the concrete wordnet class (e.g., *Actor*, *Movie*), the lowest is Wikipedia type (e.g., *Indian film Actor*). For each wordnet class, there may exist one or more entities. Additionally, an entity can also belong to one or more wordnet classes.

Definition 2. ***Heterogeneous information network (HIN)*** *[14]. Heterogeneous information network is defined as a directed graph* $G = (V, E)$ *with an object type mapping function* $\varphi : V \rightarrow \mathcal{A}$ *and a link type mapping function* $\psi : E \rightarrow \mathcal{R}$*, where* V*,* E*,* $\mathcal{A}$ *and* $\mathcal{R}$ *denotes object set, link set, object type set and relation type set, respectively, and the number of object types* $|\mathcal{A}| > 1$ *or the number of relation types* $|\mathcal{R}| > 1$*.*

Definition 3. ***Meta path*** *[14]. Meta path is a path defined on a* ***network schema*** $S = (\mathcal{A}, \mathcal{R})$ *, which is a directed graph defined over objects types* $\mathcal{A}$ *and edges types* $\mathcal{R}$*. It is denoted in the form of* $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \ldots \xrightarrow{R_l} A_{l+1}$*, which is a sequence of object types and relation types between objects, and* l *is the length of* $\mathcal{P}$*.*

In HIN, **meta path** [14] is widely used to capture the rich semantic meaning. We say a path $a_1 \xrightarrow{e_1} a_2 \xrightarrow{e_2} \ldots \xrightarrow{e_l} a_{l+1}$ between objects a_1 and a_{l+1} in network G is a **path instance** of the relevance path $\mathcal{P}$, if $\forall i$, $\phi(a_i) = A_i$ and each link $e_i = \langle a_i, a_{i+1} \rangle$ belongs to the relation R_i in $\mathcal{P}$, denoted as $p \in \mathcal{P}$. If the relation $\mathcal{R}$ is symmetric, we say a meta path is symmetric. e_i^{-1} denotes the opposite direction of the edge labeled e_i and holds for $a_{i+1} \xrightarrow{e_i^{-1}} a_i$.

Since KG contains different types of objects (i.e., subject and object) and links (i.e., predicate), KG is a natural HIN. In Fig. 1(a), $actedIn$ and $directed$ are two kinds of links types, actor and film director are different object types. Toby Kebbell and Martin McCann belong to actor class. Toby Kebbell and Nigel Havers are not only the instances of actor class but also included in the actors who acted in movies directed by Steven Spielberg. In order to distinguish the two kinds of sets, we call the latter as the fine grained set and the former as the coarse grained set in this paper.

$Person \xrightarrow{actedIn} Movie \xrightarrow{directed^{-1}} Person$ is a meta path between Toby Kebbell and Steven Spielberg shown by the dashed line in Fig. 1(a), $Toby\ Kebbell \xrightarrow{actedIn} War\ Horse\ (film) \xrightarrow{directed^{-1}} Steven\ Spielberg$ is a path instance of the meta path, $directed^{-1}$ is the opposite direction of the edge $directed$. Another meta path connecting Person and Person types is $Person \xrightarrow{actedIn} Movie \xrightarrow{directed^{-1}} Person \xrightarrow{directed} Movie \xrightarrow{actedIn^{-1}} Person$. Obviously, these two meta paths illustrate different relations of person type entities. Specifically, the first path means that the first person acted in the movies directed by the second person, while the second path means these two persons acted in the movies directed by the same director. The first path is asymmetric while the second one is symmetric. The symmetric path reveals that these two persons have the potential common feature expressing the semantic meaning of the special class. This also uncovers the intrinsic characteristics of the proposed method for ESE.

Different paths reveal different relations among entities. In order to find the entities with common features that facilitate accurate entity expansion, we can discover them using meta path based similarity measures. The entity with larger similarity has much more common characteristics than that with low similarity. Currently, there have been several path based similarity measures in HIN. For example, Lao et al. [22] devise an algorithm of Path Constrained Random Walk (PCRW) to measure the similarity between any types of entities. Sun et al. [14] propose PathSim to calculate the similarity between the same types of entities. Recently, Shi et al. [23] propose HeteSim to measure the

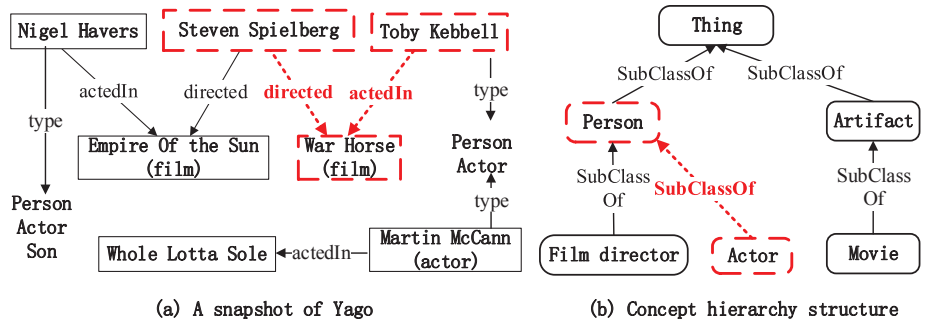


Fig. 1. A snapshot of Yago with concept hierarchy structure.

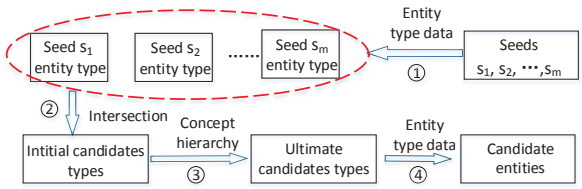


Fig. 2. The procedure of candidate entities extraction.

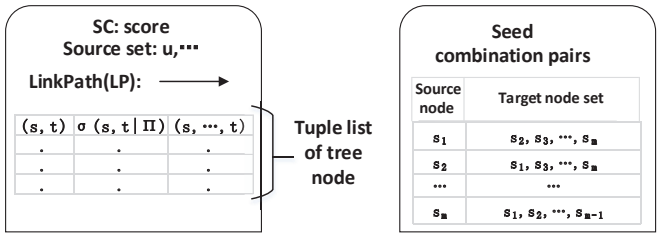


Fig. 3. Notation of data structure and seed combination pairs.

relevance of any type of entities based on symmetric or asymmetric paths. For the ESE issue, we would evaluate their performance using different similarity measures.

4 The Proposed Method

In order to solve the ESE problem with knowledge graph, we propose a novel approach called *M*eta *P*ath based *E*ntity *S*et *E*xpansion (MP_ESE). As we have mentioned, KG is a natural HIN, so we employ the widely used meta path in HIN to exploit the potential common features of seeds and automatically find the useful meta paths between seeds. Then we employ heuristic learning and PU learning methods to assign the importance to extracted meta paths and utilize the weighted meta paths to expand entities.

4.1 Candidate Entities Extraction

Because the number of entities in knowledge graph is extremely huge, it is unpractical and unreasonable to compute the similarity between each entity and given seeds. In order to reduce the number of candidate entities, we design a strategy, which leverages concept hierarchy structure introduced in Section 3, to get a proper subset of candidate entities from knowledge graph. Specifically, it includes the following four steps as shown in Fig. 2. Step 1 obtains entity types of each seed. Step 2 generates the initial common candidates types by the intersection operation. Step 3 filters the initial candidates types with the concept hierarchy structure. Step 4 extracts candidate entities satisfying the ultimate candidates types.

In order to clearly illustrate the process of candidate entity extraction, we take three seeds used in elaborating algorithm SMPG as an example, which include Toby Kebbell, Nigel Havers and Harrison Ford. Table 1 shows the detailed results. Note that the first column is the given seeds. The second column denotes the corresponding entity types of the given seeds and they are {person, actor}, {son, person, actor} and {environmentalist, aviator, carpenter, person, actor}, respectively. The third column implies the intersection of all entity types called the initial candidates types, that is {person, actor}. Some candidates types may be too general, which makes the number of candidate entities large. Therefore, we filter some candidates types using concept hierarchy structure as shown in Fig. 1(b). Particularly, we choose the most specific class closest to the bottom of the structure as the ultimate candidates type shown in the fourth column. Finally, the fifth column provides the final candidates which are extracted from Yago according to the ultimate type.

4.2 Seed-based Meta Path Generation

In order to automatically discover meta paths between seeds, we design the Seed-based Meta Path Generation algorithm (SMPG). The basic idea is that SMPG begins to search the KG from all seeds and finds important meta paths that connect certain number of seed pairs, and the meta paths can reveal the implicit common characteristics of seeds.

The process of meta path generation is traversing the KG in deed, and thus a novel tree structure is introduced in SMPG. SMPG works by expanding the tree structure and Fig. 3(a) shows the data structure of each tree node, which stores a tuple list of entity pairs with similarity values and the sets of being visited entities. In particular, the tuple form of the list is $\langle(s,t),\sigma(s,t|\prod),(s,\cdots,t)\rangle$, where (s,t) denotes the source node and target node of the current path $\prod$. The score *SC* is used to measure the importance of the tree node or discovered path. Each tree edge denotes the link type between entities. The root node of the tree contains

TABLE 1
An example of candidate entities extraction

Seed entity	Entity type	Intersection	Ultimate type	Candidates
Toby Kebbell Nigel Havers Harrison Ford	{person, actor} {son, person, actor} {environmentalist, aviator, carpenter, person, actor}	{person, actor}	{actor}	Nancy Stafford, Pirou Weston Beggard ...

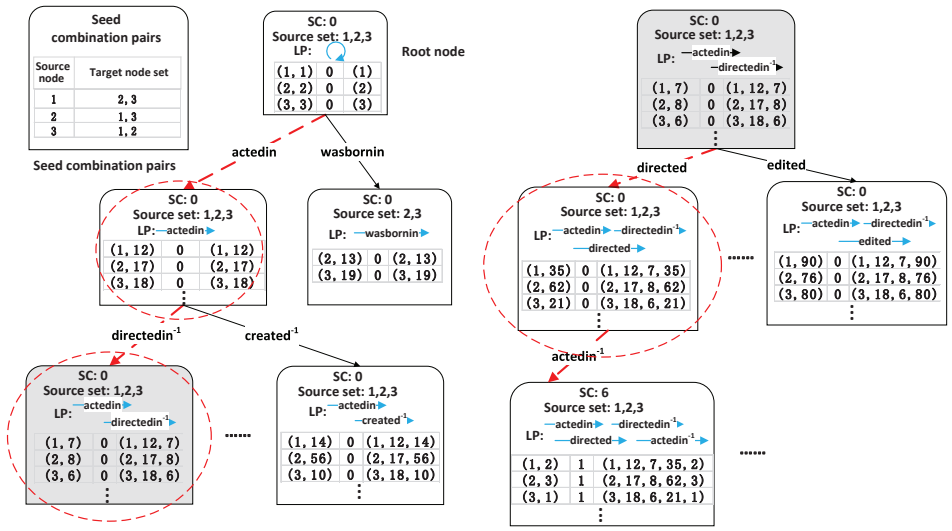


Fig. 4. Seed-based meta path generation method.

all entity pairs composed of each seed and itself. SMPG starts to expand from the root node step by step to discover important meta paths. At each step, we check whether the score SC of the tree node is larger than the predefined threshold value ν. If so, we pick out the corresponding meta path; otherwise make a move forward until the tree can not be further expanded. When moving forward, we choose the tree node with the maximum number of source set as well as the minimum number of tuples to expand, which indicates that the path of the tree node covers more seeds and has a better discriminability to get more accurate entities. Here, the reason why we design the condition of the minimum number of tuples is that it can be applied to evaluating the discriminability of the tree node. The less the number of tuples the larger the discriminability of the corresponding path of tree node. Therefore, we can greedily obtain more informative meta paths.

Specifically, in SMPG, we use a source set in the tree node to record the source nodes of all entity pairs in tuple list. The predefined threshold value ν denotes the minimum number of seed combination pairs that the important meta path connects. The more the number of the connected seed combination pairs, the more important the meta path. This also shows that more seed entities have the feature captured by the meta path, which implies that the meta path can better reveal the implicit semantic meaning of seed entities. Therefore, we can expand more entities belonging to the special class with the meta path. Supposing that a meta path only connects a pair of seed entities, which means that only the connected seed entities have the feature captured by the meta path, and other seed entities have not. Then this meta path will not represent the semantic class well and also will not be employed to expand more correct entities.

In addition, in order to prevent the repeated traversal of the nodes in SMPG, we record the nodes having been visited along the path $\prod$ in $(s, \cdots, t)$ of the tuple $\langle(s,t),\sigma(s,t|\prod),(s,\cdots,t)\rangle$. Here, $\sigma(s,t|\prod)$ is the similarity that represents whether node t is in the target node set of source node s, it is 1 if so and 0 otherwise. The target node set of each source node can be found in seed combination pairs as shown in Fig. 3(b) and each seed can be combined with the other seeds. $\sigma(s,t|\prod)$ also indicates whether the meta path connects the seed pair. And seed pairs that each meta path connects are also recorded. LP is the passing link path, the score SC of the tree node is the sum of all tuples similarities, which measures the importance of the tree node or path.

Let us elaborate the algorithm with an example shown in Fig. 4, where the set of seeds is {Toby Kebbell, Nigel Havers, Harrison Ford} marked as {1,2,3} for simplicity. All these three seeds entities are actors of the movies directed by Steven Spielberg. The set of seed combination pairs is {[1,(2,3)], [2,(1,3)], [3,(1,2)]} shown in Fig. 4. The root node of the tree contains all entity pairs composed of each seed and itself, and has $SC = 0$. The first expansion passes through two types of links: $actedIn$ and $wasBornIn$, and gets two new tree nodes. For each new tree node, SMPG records each tuple, P and SC as well as source set. At the moment, all paths do not connect any seed pairs, so we choose the tree node with the maximum number of source set as well as the minimum number of tuples to expand, which has more discriminability. Here, we choose the tree node with link $actedIn$ to expand and then get five new tree nodes. Fig. 4 only demonstrates two of them.

After the second expansion, there is still no path connect-

Algorithm 1 Seed-based Meta Path Generation Algorithm

Input: Knowledge graph G, seed set $S = \{s_1, s_2, \cdots, s_m\}$.
Output: Paths set P, seed pairs SP that each meta path connects.

1: Create the root node of the tree T;
2: $sl \Leftarrow$ link types set;
3: **while** T can be expanded **do**
4: $N \Leftarrow$ tree nodes with the maximum number of source set in T;
5: $n \Leftarrow$ tree node with the minimum number of tuples in N;
6: **for** each tuple $tp \in n$ **do**
7: get pair $tp.(s,t)$;
8: **for** each neighbor e of $tp.t$ in G **do**
9: $l \Leftarrow$ link from $tp.t$ to e;
10: **if** e not being visited and $l \in sl$ **then**
11: **if** l not in $n.child$ **then**
12: add l to $n.child$;
13: create a new child tree node tn with key $n.key + l$;
14: **end if**
15: **if** $(s,e) \in$ *seed combination pairs* **then**
16: $\sigma(s,e|\prod) \Leftarrow 1$;
17: add (s,e) to the tree node tn;
18: **else**
19: $\sigma(s,e|\prod) \Leftarrow 0$;
20: **end if**
21: add e to the visited set $(s, \cdots, t)$;
22: insert tuple $\langle (s,e), \sigma(s,e|\prod), (s,e,\cdots,t)\rangle$ into tree node;
23: add $tp.s$ to the source set of tree node tn;
24: update the SC of tree node tn;
25: **end if**
26: **end for**
27: **end for**
28: **for** each node en in T **do**
29: **if** $en.SC > threshold\ \nu$ **then**
30: add meta path $\prod$ of en into P;
31: add the seed pairs that $\prod$ connects into SP;
32: **end if**
33: **end for**
34: **end while**
35: **return** P, SP;

ing seed pairs. Then we continue to choose the tree node with the maximum number of source set and the minimum number of tuples to expand, and we update the corresponding values. Except seeds 1,2 and 3, the other marked entities such as 35, 62 denote those being visited in-between entities of the path. After several expansions, a length-4 path $Actor \xrightarrow{actedIn} Movie \xrightarrow{directed^{-1}} Person \xrightarrow{directed} Movie \xrightarrow{actedIn^{-1}} Actor$ is found shown by the dash line in Fig. 4. And we continue to repeat the process until the condition is satisfied or the tree can not be further expanded.

We present the detailed steps of SMPG in Algorithm 1. Firstly, we create the root node of the tree in Step 1 and give some predefined constants in Step 2. Then we expand the tree and find the important meta paths in Steps 3-34. At each expansion, we choose the tree node with the maximum number of seeds as well as the minimum number of tuples in Steps 4,5. Step 10 judges whether the link is in the set of the given link type, whether the neighbor node isn't visited before. If so, we make an expansion and examine whether the entity pair is in seed combination pairs in Step 15. If so, we record the connected entity pair. And we insert the new tuple into the corresponding tree node in Step 22. Meanwhile Step 23 adds the source node of the entity pair to the source set of the tree node and Step 24 updates SC. Steps 28-33 get the expected meta paths and the corresponding seed pairs.

4.3 Weight Learning of Meta Paths

SMPG discovers the important meta paths P, but the importance of each meta path is different for the further entity set expansion. In order to effectively combine these meta paths to obtain ESE model, we should devise an appropriate weight learning method. Because there is only a small amount of positive examples and a great deal of unlabeled examples, without negative examples and effective methods for negative data automatic selection, we can not use the traditional supervised learning method to train the model. To deal with the issue, we design two weight learning solutions, namely heuristic learning method and PU learning method.

The first solution is the heuristic learning method. Intuitively, the importance of meta path should be related to the number of seed pairs connected by the meta path. The more seed pairs the meta path connects, the more important it is. Thus, we consider the ratio of SP_k and $m*(m-1)$ to be the weight w'_k of meta path p_k ($p_k \in P$), where SP_k is the number of seed pairs connected by the meta path P_k, $m*(m-1)$ denotes the total number of seed pairs and m is the number of seeds. Here, we call this method as heuristic learning method. In order to normalize w'_k, we define the final weight as follows:

$$w_k = \frac{w'_k}{\sum_{k=1}^{l} w'_k}, \tag{1}$$

where l is the number of meta paths P.

With the w_k, we can combine meta paths to get the following ranking model.

$$R(c_i, S) = \frac{1}{m}\sum_{j=1}^{m}\sum_{k=1}^{l} w_k \cdot r\{(c_i, s_j)|p_k\}, \tag{2}$$

where c_i denotes the ith candidate entity, n is the number of candidates. $S = \{s_1, s_2, \cdots, s_m\}$ is the set of seeds, $r\{(c_i, s_j)|p_k\}$ denotes whether the path p_k connects c_i and s_j, it is 1 if connected and 0 otherwise.

Here, we use an example to justify the reasonability of the heuristic weight. The seeds are still Toby Kebbell, Nigel Havers and Harrison Ford, marked as 1, 2 and 3. SMPG finds the meta path $Person \xrightarrow{actedIn} Movie \xrightarrow{directed^{-1}} Person \xrightarrow{directed} Movie \xrightarrow{actedIn^{-1}} Person$, denoted as p_1, and meta path $Person \xrightarrow{actedIn} Movie \xrightarrow{edited^{-1}} Person \xrightarrow{edited} Movie \xrightarrow{actedIn^{-1}} Person$, denoted as p_2. p_1 connects seed pairs $(1,2), (1,3)$ and $(2,3)$ while p_2 only connects seed pairs $(1,2)$. We can find that the number of seed pairs connected by p_1 is larger than that connected by p_2. Hence, the meta path p_1 is more important than p_2. p_1 reveals the common feature of all seeds and could denote the semantic meaning of the particular class. By comparison, p_2 only reveals the characteristic of partial seeds and could not better represent the class.

The second solution is the PU learning (positive and unlabeled learning) method, which is a class of semi-supervised learning algorithms. Given a set of positive examples of a particular class and a set of unlabeled examples (containing hidden positive and negative cases), PU learning aims to building a classifier using these positive

and unlabeled examples for classifying the unlabeled data [10]. One kind of PU approaches is based on a two-step strategy: (1) identifying a set of reliable negative documents from the unlabeled set; and (2) building a classifier using EM or SVM iteratively [57]. There is also some work learning a traditional classifier from non-traditional input based on the given assumption [58]. As described in [10], the set expansion problem can be exactly mapped into PU learning, with seeds and candidates as positive and unlabeled examples respectively.

Here, we adopt a classical PU learning method presented by Elkan et al. [58] to solve the problem of ESE. The method is applied in document classification in [58] and has never been adopted to solve the ESE problem so far. There is a difference between the scenario of ESE considered here and the scenario of document classification considered in [58]. The scenario here is that we treat the seed pairs as positive data while the pairs of candidate entities and seed entities could be seen as unlabeled data. The scenario of [58] is that both the positive and the negative are single documents. In addition, it can flexibly choose any traditional classifier for PU learning, so that we can choose the suitable one for better results. Finally, we select Adaboost as base estimate model to do PU learning since it can change the distribution of training data to increase the importance of positive data and achieve a good classification result.

This method proposed by Elkan et al. [58] is based on the assumption of the "selected completely at random". In this paper, we randomly select the seed entities from all positive examples, which is consistent with the assumption. Under the assumption, the probability that a positive example is labeled is a constant. The constant plays a great role in obtaining a traditional classifier. The paper [58] introduces three kinds of estimation methods and states that the constant can be estimated using a trained non-traditional classifier and a validation set of examples. Actually, more strategies can be designed in leveraging the information above to obtain the estimate of the constant. Here, we design a new strategy, that is the median of the nontraditional classifier for the sampled positive examples, to estimate the constant. As we all know, the median is not affected by the maximum or minimum value of the distribution sequence, which can improve the representativeness of the median to the distribution sequence to a certain extent.

We can compute relevance between each candidate entity and each seed using the ranking model in Eq. 2, and then rank all candidate entities.

4.4 Algorithm Framework and Analysis

We have introduced the detailed information about MP_ESE above, which includes the following three steps. Firstly, we design a strategy of extracting candidate entities. Secondly, we develop an algorithm called Seed-based Meta Path Generation (SMPG) to automatically discover important meta paths between seeds. Finally, we employ heuristic learning method and PU learning method to combine extracted meta paths for the further ESE.

For MP_ESE, the time complexity includes two parts: (1) automatically finding meta paths: the time complexity of finding important meta paths between seeds is related to m and $\bar{d}$, where $\bar{d}$ is the average degree of the entities in knowledge graph and m is the number of seeds. (2) Ranking candidate entities: the time complexity of ranking candidate entities is $O(mnk + l\bar{M}^2)$. The first part $O(mnk)$ is the time complexity of calculating the similarity for all candidate entities and the seeds set, where m is the number of seeds, n is the number of candidate entities and k is the number of the discovered important meta paths. The second part $O(l\bar{M}^2)$ is the time complexity of building the connection matrix based on the given meta path, where l is the length of the given meta path and $\bar{M}$ represents the average number of all entities connected by different relation R_i of the given meta path $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \dots \xrightarrow{R_l} A_{l+1}$, that is the average dimensionality of adjacent matrix corresponding to different relation R_i.

5 Experiments

5.1 Dataset

As a typical KG, Yago is a huge semantic knowledge graph derived from Wikipedia, WordNet and GeoNames [11]. Currently, it has knowledge about more than 10 million entities and contains more than 120 million facts [2]. We adopt "yagoFacts", "yagoSimpleTypes" and "yagoTaxonomy" parts of this dataset to conduct experiments, of which the "yagoFacts" contains 35 relationships, more than 1.3 million entities of 3455 classes. Table 2 provides the data description and statistics.

We choose six representative expansion tasks to evaluate the performance of MP_ESE comprehensively. The classes used in these tasks are summarized as follows: actors of the movies directed by Steven Spielberg, softwares of the companies located in Mountain View of California, movies whose director won National Film Award, scientists of the universities located in Cambridge of Massachusetts, writers being graduated from the universities in New York, and married actors who won Emmy Award and their spouses are also actors. The six classes are written as Actor*, Software*, Movie*, Scientist*, Writer* and Married_actor*. The real number of positive entities in these six classes are 112, 98, 653, 202, 60 and 193, respectively. The way of obtaining the positive entities will be introduced in Section 5.2. The number of candidate entities are 41896, 2337, 44887, 12897, 27319 and 44887, respectively.

5.2 Criteria

We employ two popular criteria of precision-at-k ($p@k$) and mean average precision (MAP) to evaluate the performance of our approach. $p@k$ is the percentage of top k results that belong to correct entities. Here, they are $p@30$, $p@60$ and $p@90$. MAP is the mean of the average precision (AP) of the $p@30$, $p@60$ and $p@90$. $AP = \frac{\sum_{i=1}^{k} \text{p@}i \times rel_i}{\#\text{ of correct entities}}$, where rel_i equals 1 if the result at rank i is correct entities and 0 otherwise.

In order to obtain the correct answers for comparing the results, we extract correct entities from yagoFacts part of Yago dataset according to the semantic meaning

2. http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/

TABLE 2
Description of the data.

Data	Template of triples	# triples	# entities/types
yagoFacts	< entity relatinship entity >	4,484,914	1,369,931
yagoSimpleTypes	< entity rdf:type wordnet_type >	5,437,179	6,543
yagoTaxonomy	< wordnet_type rdfs:subclassof wordnet_type >	69,826	388,606

of expansion tasks. The yagoFacts part describes all facts between entities. We can extract correct entities satisfying the semantic meaning of the expansion task through leveraging all these facts relationships between entities. For example, Actor* task refers to the actors who act in the movies directed by Steven Spielberg. And there exist two kinds of facts in yagoFacts part, one satisfies the form of $< Person, actedIn, Movie >$, the other satisfies the form of $< Steven\ Spielberg, directed, Movie >$. Through the common entity with object type as Movie, we can connect the two kinds of facts and obtain the following tuples of $< Person, actedIn, Movie, directed^{-1}, Steven\ Spielberg >$, then we can filter out the actor entities who act in the movies directed by Steven Spielberg, which are the correct answers for comparing the results. In the same way, we can obtain correct answers for other tasks.

5.3 Effectiveness Experiments

In this section, we will validate the effectiveness of MP_ESE on entity set expansion. The MP_ESE with heuristic learning method and PU learning method are denoted as MP_ESE_He and MP_ESE_PU, respectively. Since there are no direct solutions for ESE on KG, we design some baselines which are summarized as follows.

- Link-Based. According to the pattern-based methods in text or Web environment, we only consider 1-hop link of an entity, denoted as Link-Based.
- Nearest-Neighbor. Inspired by QBEES [50], [51], we consider 1-hop link and 1-hop entity at the same time, called Nearest-Neighbor.
- PCRW. Based on the path constrained random walk [22], we compare with length-1, length-2, length-3, length-4 paths, denoted as PCRW-1, PCRW-2, PCRW-3 and PCRW-4, respectively.

For each class introduced above, we randomly take three seeds from the entity set to conduct an experiment. We run algorithms 30 times and report the average results. In MP_ESE, we empirically set the predefined threshold value ν to be $m*(m-1)/2+1$, which can guarantee that the path connects half number of seeds or more, m is the number of seeds. As pointed out in [22], the number of possible meta paths for a given heterogeneous graph grows exponentially with the length of meta path, and the execution time also exponentially increases. Sun et al. [14] also point out that a very long path is not very meaningful. In order to balance between quality and time, we fix the max path length to 4. The optimal parameters are set for other baselines.

The overall results of entity set expansion effectiveness are given in Fig. 5. From Fig. 5, we can see that our approaches (MP_ESE_PU and MP_ESE_He) achieve better performance than other methods on almost all conditions. However, Link-Based has very bad performance. We think the reason is that the 1-hop link contains few semantic meaning and can not further distinguish the characteristics of the fine grained class, but our approaches can distinguish them well. On the Software* task, MP_ESE_PU, MP_ESE_He, PCRW-2 and PCRW-3 have close performance. This is because Software* is an overlapped class and has another class label depicted by length-2 path $Software \xrightarrow{created^{-1}} Company \xrightarrow{created} Software$. The performance of PCRW-4 is also good on the Software* task. Although PCRW-4 sometimes has good performance, PCRW-4 needs to find all length-4 meta paths between seed entities, which could contain meaningless or noisy paths. Just as we have illustrated above, the number of paths grows exponentially with the length of the path. Therefore, it is impractical to find all meta paths between seed entities especially for the large-scale KG dataset and PCRW-4 is not beneficial for expansion tasks.

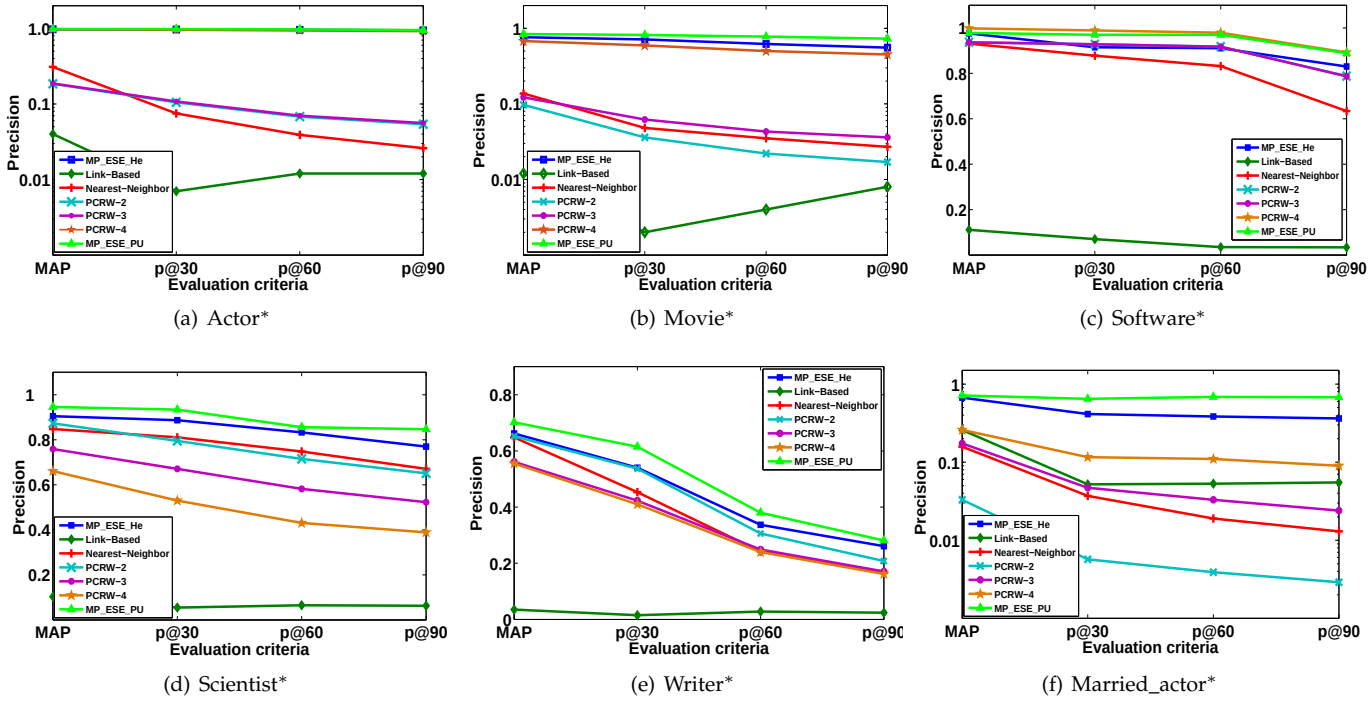
From Fig. 5, we can also discover that MP_ESE_PU always performs better than MP_ESE_He on all tasks, which indicates that the PU learning is better than the heuristic method to assign the weights of paths. The reason is that the PU learning can discover more intrinsic relationship of path features for ESE and construct more appropriate model than the heuristic method. In all, MP_ESE_PU has almost the best performance because it not only employs the way of the meta path, which can capture the subtle semantic meaning between entities, but also adopts the PU learning method to automatically learn the weights of meta paths.

In order to intuitively observe the effectiveness of discovered meta paths, Tables 3 and 4 depict the top 3 meta paths, heuristic weight and PU learning weight on Actor* and Software* tasks, respectively. We observe that these meta paths reveal some common characteristics of actor or software class. Both weight learning methods give the largest importance for the first path, which indicates the effectiveness of the proposed methods. For Actor*, the first meta path indicates that actors act in movies directed by the same director, which shows that SMPG can effectively mine the most important semantic meaning of Actor*. The second and the third meta paths imply that some actors act in movies edited or composed by the same person. For Software*, we can find that the weights of the first and second meta paths are both relatively large, which indicates that software is an overlapped class. In the end ,through leveraging the important meta paths discovered by SMPG, we can find other entities belonging to the same class with seeds.

5.4 Impact of Seed Size

To evaluate the impact of seed size on the performance, we conduct relevant experiments in the range of seed size from 2 to 6 on Actor* task. For each seed size, we randomly

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2018.2805366, IEEE Transactions on Big Data



(a) Actor* (b) Movie* (c) Software*

(d) Scientist* (e) Writer* (f) Married_actor*

Fig. 5. Results of effectiveness experiments on six tasks.

TABLE 3
Top 3 meta paths, heuristic and PU weights for Actor* task.

meta path	heuristic weight	PU weight
$\text{Person} \xrightarrow{actedIn} \text{Movie} \xrightarrow{directed^{-1}} \text{Person} \xrightarrow{directed} \text{Movie} \xrightarrow{actedIn^{-1}} \text{Person}$	0.2180	0.2082
$\text{Person} \xrightarrow{actedIn} \text{Movie} \xrightarrow{writeMusicFor^{-1}} \text{Person} \xrightarrow{writeMusicFor} \text{Movie} \xrightarrow{actedIn^{-1}} \text{Person}$	0.1495	0.1561
$\text{Person} \xrightarrow{actedIn} \text{Movie} \xrightarrow{edited^{-1}} \text{Person} \xrightarrow{edited} \text{Movie} \xrightarrow{actedIn^{-1}} \text{Person}$	0.1476	0.1399

TABLE 4
Top 3 meta paths, heuristic and PU weights for Software* task.

meta path	heuristic weight	PU weight
$\text{Software} \xrightarrow{created^{-1}} \text{Company} \xrightarrow{islocatedin} \text{city} \xrightarrow{islocatedin^{-1}} \text{Company} \xrightarrow{created} \text{Software}$	0.5468	0.6125
$\text{Software} \xrightarrow{created^{-1}} \text{Company} \xrightarrow{created} \text{Software}$	0.3571	0.3571
$\text{Software} \xrightarrow{created^{-1}} \text{Company} \xrightarrow{owns} \text{Property} \xrightarrow{owns^{-1}} \text{Company} \xrightarrow{created} \text{Software}$	0.0167	0.0152

select the same scale seeds from the entity set to conduct an experiment. We run our algorithm 30 times and record the maximum, the minimum and the average results, which are demonstrated in Table 5.

It can be seen that the performance has an improvement with the increasing of seed size. The performance with 2 seeds is the lowest, since 2 seeds do not contain plenty of information and may have several class labels. The performance with 3 seeds has been good enough. When the number of seeds is larger than 3, the improvement is minor. Therefore, we employ 3 seeds in other experiments. In order to better clarify the change trend of performance along the seed size, Fig. 6 shows the mean results of different criteria. We can observe that the performances of different criteria have an increasing trend and finally converge. In all, the proper seed size should be determined. Besides, there is a big difference between the maximum and minimum precisions because of the random seed sets. Then we should find out the impact of different seed combinations.

5.5 Effect of Seed Combination

In order to study the effect of different seed combinations on the performance, we run algorithms 30 times and randomly select three entities as seeds at each time. Fig. 7 records the MAP values and the horizontal coordinate denotes 30 different seed combinations. In Fig. 7, we sort the results in ascending order so as to show them clearly.

From Fig. 7, we can see that the seed combination has an effect on the expansion performance. There is a wide variation between the maximum and minimum performances

TABLE 5
The impact of seed size for Actor* task.

Seed size	p@30			p@60			p@90			MAP		
	max	min	mean	max	min	mean	max	min	mean	max	min	mean
2	1.0	0.067	0.858	1.0	0.117	0.834	1.0	0.111	0.811	1.0	0.110	0.900
3	1.0	0.433	0.970	1.0	0.333	0.959	1.0	0.256	0.945	1.0	0.427	0.976
4	1.0	0.733	0.977	1.0	0.517	0.957	1.0	0.4	0.938	1.0	0.852	0.988
5	1.0	0.967	0.998	1.0	0.95	0.996	1.0	0.767	0.988	1.0	0.967	0.997
6	1.0	1.0	1.0	1.0	0.933	0.996	1.0	0.867	0.991	1.0	0.987	0.999

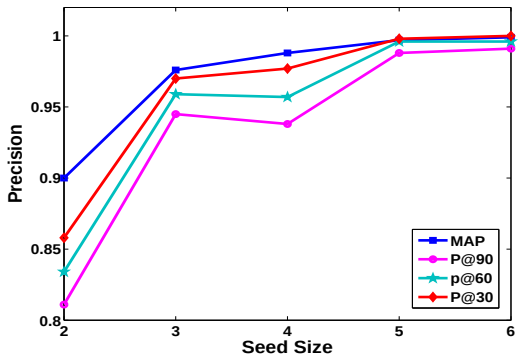


Fig. 6. Precision change of MP_ESE along seed size on Actor*.

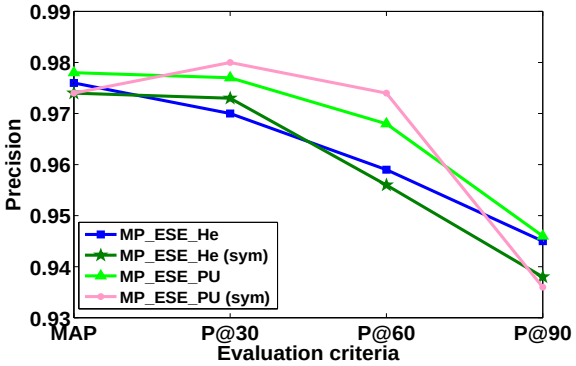


Fig. 8. Precision comparison of two methods on Actor*.

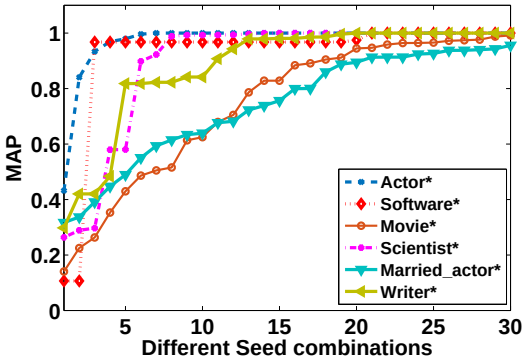


Fig. 7. Precision change of different seed combinations.

on each task. The worst even reaches to 0.106 on Software* task, while the best is close to 1. Therefore, we can confirm that seed composition has a significant impact on the quality of expansion. It is of importance to choose a good seed combination and the inferior should not be employed, which is left for the further research.

5.6 Influence of Weight

To demonstrate the influence of the weight on the performance, We consider 4 different weights which include heuristic weight, random weight, average weight and PU weight. On six tasks, we conduct experiments 30 times and randomly choose three entities as seeds at each time. Table 6 records the average results on six tasks.

We can observe that the heuristic weight and PU weight have better performance than average and random weights on the whole, which suggests that the importance of meta paths is different and some paths can better reflect the implicit features of seeds than others. And the effect of the weight is also different for different tasks. For Software* task, weight has a minor effect on performance, because the total number of meta paths is 5 and it is an overlapped class. The PU weight has the best performance, which shows that the semi-supervised learning algorithm can deeply explore the semantic meaning between seeds and learn the more appropriate weight.

5.7 Impact of Symmetric Meta Path

In this subsection, we study the impact of symmetric meta path on the performance. For the ESE issue, we think that the symmetric meta path is very significant and has a major impact on the effectiveness, while the asymmetric meta path has a minor effect. In order to validate this point, we only consider the symmetric meta path for MP_ESE with heuristic weight and PU weight to conduct relevant experiments, which are denoted as MP_ESE_He (sym) and MP_ESE_PU (sym), respectively. MP_ESE_He and MP_ESE_PU consider symmetric and asymmetric meta paths simultaneously. We conduct experiments 30 times and randomly choose three seeds at each time. Fig. 8 records the average values of MAP, p@30, p@60 and p@90 on Actor* task.

From Fig. 8, we can see that the performance difference of two methods is not very obvious. It turns out that the symmetric meta paths make a great contribution to the performance of entity set expansion and the asymmetric meta paths have a little effect. The great contribution of symmetric meta paths demonstrates that the seeds have the same common features showing the special semantic meaning class. The experiment also reveals the intrinsic characteristics of the proposed method for ESE.

TABLE 6
The impact of different weights on the performance. A bigger value indicates a better performance.

Class	Heuristic weight				Average weight				Random weight				PU weight			
	p@30	p@60	p@90	MAP	p@30	p@60	p@90	MAP	p@30	p@60	p@90	MAP	p@30	p@60	p@90	MAP
Actor*	0.970	0.959	0.945	0.976	0.948	0.934	0.917	0.958	0.941	0.915	0.889	0.949	0.977	0.968	0.946	0.978
Software*	0.915	0.911	0.830	0.926	0.911	0.908	0.824	0.925	0.918	0.903	0.812	0.930	0.969	0.969	0.889	0.978
Movie*	0.711	0.620	0.554	0.760	0.639	0.530	0.461	0.710	0.480	0.414	0.369	0.554	0.816	0.775	0.728	0.838
Scientist*	0.887	0.833	0.770	0.905	0.822	0.743	0.665	0.871	0.546	0.486	0.429	0.627	0.92	0.851	0.834	0.935
Writer*	0.540	0.337	0.261	0.662	0.27	0.195	0.173	0.275	0.215	0.165	0.137	0.275	0.613	0.380	0.274	0.685
Married_actor*	0.414	0.385	0.363	0.671	0.305	0.307	0.302	0.401	0.306	0.288	0.290	0.360	0.648	0.687	0.680	0.715

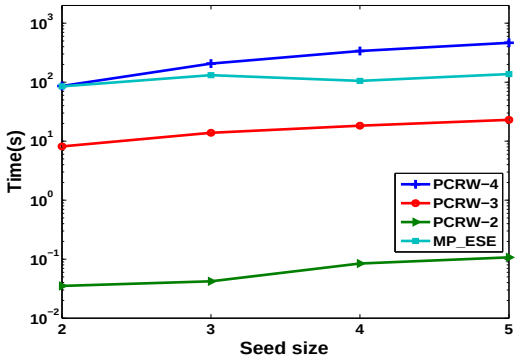


Fig. 9. Running time of finding meta path with different seed sizes on Actor*.

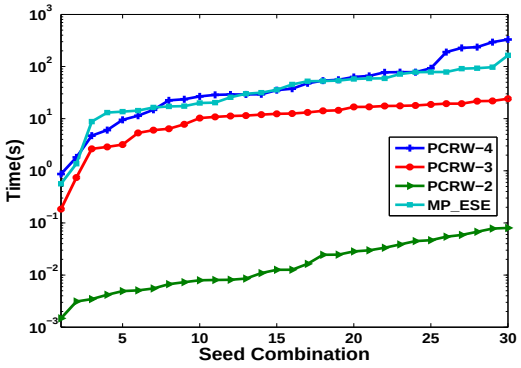


Fig. 10. Running time of finding meta path with different seed combinations on Actor*.

5.8 Efficiency

In this subsection, we evaluate how seed size and seed combination affect the efficiency of finding meta paths. For the efficiency evaluation of seed size, we conduct experiments by varying the seed size from 2 to 5 on Actor* task. For each seed size, we randomly select the same scaled seeds to run the algorithm once. We repeat the experiment 20 times and the average is reported in Fig. 9. To examine the impact of different seed combinations, we run algorithms 30 times and randomly select three entities as seeds at each time. Fig. 10 demonstrates the results after sorting the running time in ascending order on Actor* task. Here, we do not list the PCRW-1 method since there are not length-1 paths. We do not list Linked-Based and Nearest-Neighbor approaches as well, because these two approaches do not belong to the methods of finding meta paths.

From Fig. 9, we can observe that the running time of finding paths has a rising trend with the increasing of seed size. From Fig. 10, we see that the running time of the same method varies based on different seed combinations. The time of different methods is also different based on the same seed combination. In Fig. 9 and Fig. 10, we can also see that our method is much faster than the PCRW-4. Although it is slower than PCRW-2 and PCRW-3, the effectiveness of our method is better than them. In summary, both seed size and seed combination have the influences on the efficiency of finding meta paths. Therefore, we should choose the proper seed size and seed combination so as to get a better performance.

5.9 Performance Study on Different Similarity Measures

When ranking candidate entities, we need calculate the similarity between candidates and seeds. Based on different similarity measures, the order of candidates should be different. In order to study the influence of similarity measures, we conduct relevant experiments using 3 different measures. The first is denoted by Reachability, which means whether the seeds and candidates are connected by the given meta path or not. The second means the number of path instances satisfying the given meta path between seed and candidate entities, denoted by Pathcount. The third is the reachable probability, written as PCRW. The three kinds of different measures are combined with MP_ESE with heuristic weight and PU weight, which can be denoted by MP_ESE_He with Reachability, MP_ESE_PU with Reachability, MP_ESE_He with Pathcount, MP_ESE_PU with Pathcount, MP_ESE_He with PCRW and MP_ESE_PU with PCRW, respectively.

We run algorithms 30 times and randomly choose three entities as seeds at each time. Fig. 11 depicts the average values of MAP, $p@30$, $p@60$ and $p@90$ on Actor* task. It is obvious that the performance of MP_ESE_PU with Reachability and MP_ESE_He with Reachability are best among three measures. The reason lies in the different semantic meaning of similarity measures. For entity set expansion issue, we pay more attention to the relation semantic between seed and candidate entities rather than the reachable probability or the number of paths between them. The relation semantic can be revealed well through the Reachability. For instance, there exists a path $Actor \xrightarrow{actedIn} Movie \xrightarrow{directed^{-1}} Person \xrightarrow{directed} Movie \xrightarrow{actedIn^{-1}} Actor$ between two

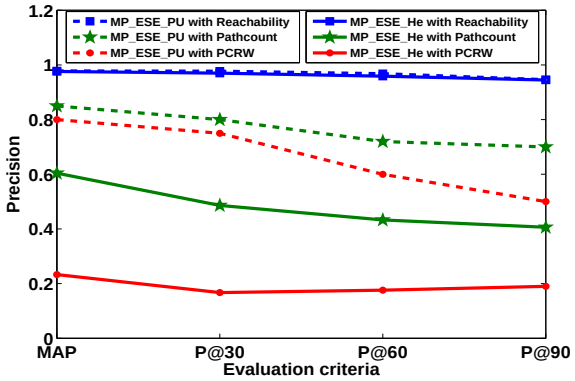


Fig. 11. Performance of MP_ESE of ranking candidates with different measures on Actor*.

actors entities, which illustrates the relation semantic that these two actors act in movies directed by the same director. If two actors can not be connected by the meta path, then these two actors have not the relation semantic and do not belong to the particular class. The similarity measure of the Reachability can illustrate whether the candidates and seeds have the same semantic meaning. It can also further demonstrate whether the candidates and seeds belong to the same class. Thus, we choose the Reachability for ESE issue. In addition, we can also see that MP_ESE_PU has better performance than MP_ESE_He on three different measures, which further illustrates that PU learning weight method is more effective than heurictic weight method.

6 Conclusion

In this paper, we study the problem of entity set expansion in knowledge graph. We model knowledge graph as a heterogeneous information network and propose a *M*eta *P*ath based *E*ntity *S*et *E*xpansion approach called MP_ESE, which employs the meta path to exploit the implicit common features of seeds. In order to automatically find the important meta paths between seeds, MP_ESE designs a novel algorithm called SMPG. And then we design the heuristic strategy and PU learning method to assign the importance of meta paths. MP_ESE utilizes the weighted meta paths to expand entities. Experiments on Yago show the effectiveness and efficiency of MP_ESE. In addition, we evaluate the influences of seed size, seed combination, weight, symmetric meta path and similarity measure on the expansion performance.

Through extensive experiments, we find that the proposed SMPG can discover the potential common features shared by the seeds, which can reveal the semantic meanings of the expansion class. We also find that the presented MP_ESE can achieve very good performance on general expansion tasks. Even on those overlapped classes expansion tasks, our method is also suitable from the perspective of the balance between effectiveness and efficiency. In the future, we will study how to select the optimal seed combination and how to determine the proper seed size. We will also examine the other weight learning method so as to obtain the more appropriate weight for the meta path.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No. 61375058), National Key Basic Research and Department (973) Program of China (No. 2013CB329606), and the Co-construction Project of Beijing Municipal Commission of Education.

References

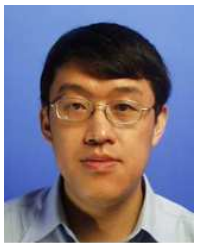
[1] W. W. Cohen and S. Sarawagi, "Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods," in *KDD*. ACM, 2004, pp. 89–98.
[2] P. Pantel and D. Lin, "Discovering word senses from text," in *KDD*. ACM, 2002, pp. 613–619.
[3] J. Hu, G. Wang, F. Lochovsky, J.-t. Sun, and Z. Chen, "Understanding user's query intent with wikipedia," in *WWW*. ACM, 2009, pp. 471–480.
[4] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li, "Context-aware query suggestion by mining click-through and session data," in *KDD*. ACM, 2008, pp. 875–883.
[5] L. Zhang and B. Liu, "Entity set expansion in opinion documents," in *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*. ACM, 2011, pp. 281–290.
[6] Y. He and D. Xin, "Seisa: set expansion by iterative similarity aggregation," in *WWW*. ACM, 2011, pp. 427–436.
[7] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *ICDM*. IEEE, 2007, pp. 342–350.
[8] ——, "Iterative set expansion of named entities using the web," in *ICDM*. IEEE, 2008, pp. 1091–1096.
[9] L. Sarmento, V. Jijkuon, M. de Rijke, and E. Oliveira, "More like these: growing entity classes from seeds," in *CIKM*. ACM, 2007, pp. 959–962.
[10] X.-L. Li, L. Zhang, B. Liu, and S.-K. Ng, "Distributional similarity vs. pu learning for entity set expansion," in *ACL*. ACL, 2010, pp. 359–364.
[11] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *WWW*. ACM, 2007, pp. 697–706.
[12] Z. Qi, K. Liu, and J. Zhao, "Choosing better seeds for entity set expansion by leveraging wikipedia semantic knowledge," in *CCPR*. Springer, 2012, pp. 655–662.
[13] C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, "A survey of heterogeneous information network analysis," *arXiv preprint arXiv:1511.04854*, 2015.
[14] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *VLDB*, vol. 4, no. 11, pp. 992–1003, 2011.
[15] Y. Zheng, C. Shi, X. Cao, X. Li, and B. Wu, "Entity set expansion with meta path in knowledge graph," in *PAKDD*. Springer, 2017, pp. 317–329.
[16] A. Singhal, "Introducing the knowledge graph: things, not strings," *Official google blog*, 2012.
[17] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs," 2016.
[18] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, no. Preprint, pp. 1–20, 2016.
[19] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano, "Template-based question answering over rdf data," in *WWW*. ACM, 2012, pp. 639–648.
[20] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum, "Natural language questions for the web of data," in *EMNLP-CoNLL*. ACL, 2012, pp. 379–390.
[21] L. Zou, R. Huang, H. Wang, J. X. Yu, W. He, and D. Zhao, "Natural language question answering over rdf: a graph data driven approach," in *SIGMOD*. ACM, 2014, pp. 313–324.
[22] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Machine learning*, vol. 81, no. 1, pp. 53–67, 2010.
[23] C. Shi, X. Kong, Y. Huang, S. Y. Philip, and B. Wu, "Hetesim: A general framework for relevance measure in heterogeneous networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 10, pp. 2479–2492, 2014.

[24] Y. Sun, C. Aggarwal, and J. Han, "Relation strength-aware clustering of heterogeneous information networks with incomplete attributes," in *VLDB*, 2012, pp. 394–405.
[25] G. J. Qi, C. C. Aggarwal, and T. S. Huang, "On clustering heterogeneous social media objects with outlier links," in *WSDM*, 2012, pp. 553–562.
[26] X. Kong, B. Cao, and P. S. Yu, "Multi-label classification by mining label and instance correlations from heterogeneous information networks," in *KDD*, 2013, pp. 614–622.
[27] Y. Zhou and L. Liu, "Activity-edge centric multi-label classification for mining heterogeneous information networks," in *KDD*, 2014, pp. 1276–1285.
[28] B. Cao, X. Kong, and P. S. Yu, "Collective prediction of multiple types of links in heterogeneous information networks," in *ICDM*, 2014, pp. 50–59.
[29] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han, "Co-author relationship prediction in heterogeneous bibliographic networks," in *ASONAM*, 2011, pp. 121–128.
[30] M. K. NG, X. Li, and Y. Ye, "MultiRank: Co-ranking for objects and relations in multi-relational data," in *KDD*, 2011, pp. 1217–1225.
[31] D. Zhou, S. A. Orshanskiy, H. Zha, and C. L. Giles, "Co-ranking authors and documents in a heterogeneous network," in *ICDM*, 2007, pp. 739–744.
[32] M. Jamali and L. Lakshmanan, "Heteromf: recommendation in heterogeneous information networks using context dependent factor models," in *WWW*, 2013, pp. 643–654.
[33] C. Shi, C. Zhou, X. Kong, P. S. Yu, G. Liu, and B. Wang, "Heterecom: a semantic-based recommendation system in heterogeneous networks," in *KDD*, 2012, pp. 1552–1555.
[34] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *arXiv preprint arXiv:1503.00759*, 2015.
[35] Y. Sun, B. Norick, J. Han, X. Yan, P. S. Yu, and X. Yu, "Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks," *TKDD*, vol. 7, no. 3, p. 11, 2013.
[36] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *WSDM*. ACM, 2014, pp. 283–292.
[37] Y. Shi, P.-W. Chan, H. Zhuang, H. Gui, and J. Han, "Prep: Path-based relevance from a probabilistic perspective in heterogeneous information networks," *arXiv preprint arXiv:1706.01177*, 2017.
[38] M. J. Cafarella, D. Downey, S. Soderland, and O. Etzioni, "Knowitnow: Fast, scalable information extraction from the web," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. ACL, 2005, pp. 563–570.
[39] M. Pasca, "Weakly-supervised discovery of named entities using web search queries," in *CIKM*. ACM, 2007, pp. 683–690.
[40] J. J. Katz, *The Philosophy of linguistics*. Oxford University Press, 1985.
[41] J. Shen, Z. Wu, D. Lei, J. Shang, X. Ren, and J. Han, "Setexpan: Corpus-based set expansion via context feature selection and rank ensemble," in *ECML-PKDD*. Springer, 2017, pp. 288–304.
[42] B. Shi, Z. Zhang, L. Sun, and X. Han, "A probabilistic co-bootstrapping method for entity set expansion," in *COLING*, 2014, pp. 2280–2290.
[43] A. Krishnan, D. Padmanabhan, S. Ranu, and S. Mehta, "Select, link and rank: Diversified query expansion and entity ranking using wikipedia," in *International Conference on Web Information Systems Engineering*. Springer, 2016, pp. 157–173.
[44] Z. Zhang, L. Sun, and X. Han, "A joint model for entity set expansion and attribute extraction from web search queries," in *AAAI*, 2016, pp. 3101–3107.
[45] L. Bing, W. Lam, and T.-L. Wong, "Wikipedia entity expansion and attribute extraction from the web using semi-supervised learning," in *WSDM*. ACM, 2013, pp. 567–576.
[46] Z. QI, K. LIU, and J. ZHAO, "A novel entity set expansion method leveraging entity semantic knowledge," *Journal of Chinese Information Processing*, vol. 27, no. 2, pp. 1–9, 2013.
[47] K. Sadamitsu, K. Saito, K. Imamura, and G. Kikui, "Entity set expansion using topic information," in *HLT*. ACL, 2011, pp. 726–731.
[48] P. Jindal and D. Roth, "Learning from negative examples in set-expansion," in *ICDM*. IEEE, 2011, pp. 1110–1115.
[49] X. Yu, Y. Sun, B. Norick, T. Mao, and J. Han, "User guided entity similarity search using meta-path selection in heterogeneous information networks," in *CIKM*. ACM, 2012, pp. 2025–2029.
[50] S. Metzger, R. Schenkel, and M. Sydow, "Qbees: query by entity examples," in *CIKM*. ACM, 2013, pp. 1829–1832.
[51] ——, "Aspect-based similar entity search in semantic knowledge graphs with diversity-awareness and relaxation," in *CWI and IAT*. IEEE Computer Society, 2014, pp. 60–69.
[52] B. Fetahu, U. Gadiraju, and S. Dietze, "Improving entity retrieval on structured data," in *International Semantic Web Conference*. Springer, 2015, pp. 474–491.
[53] M. Kahng and S.-g. Lee, "Exploiting paths for entity search in rdf graphs," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 1027–1028.
[54] J. Chen, Y. Chen, X. Du, X. Zhang, and X. Zhou, "Seed: A system for entity exploration and debugging in large-scale knowledge graphs," in *ICDM*. IEEE, 2016, pp. 1350–1353.
[55] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*. Springer, 2007, pp. 722–735.
[56] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *SIGMOD*. ACM, 2008, pp. 1247–1250.
[57] X. Li, B. Liu, and S.-K. Ng, "Learning to identify unexpected instances in the test set," in *IJCAI*, vol. 7, 2007, pp. 2802–2807.
[58] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *KDD*. ACM, 2008, pp. 213–220.

Yuyan Zheng received the B.S. degree from the Qingdao University of Technology in 2011, the M.S. degree from the Shandong Normal University in 2015. She is currently working toward the PhD degree in the School of Computer Science at BUPT. Her research interests are in data mining and machine learning, especially heterogeneous information network studies.

Chuan Shi received the B.S. degree from the Jilin University in 2001, the M.S. degree from the Wuhan University in 2004, and Ph.D. degree from the ICT of Chinese Academic of Sciences in 2007. He joined the Beijing University of Posts and Telecommunications as a lecturer in 2007, and is a professor and deputy director of Beijing Key Lab of Intelligent Telecommunications Software and Multimedia at present. His research interests are in data mining, machine learning, and evolutionary computing. He has published more than 40 papers in refereed journals and conferences.

Xiaohuan Cao received the B.S. degree from the Beijing University of Posts and Telecommunications(BUPT) in 2015. She is currently working toward the M.S. degree in the School of Computer Science at BUPT. Her research interests are in data mining and machine learning, especially heterogeneous information network studies.



Xiaoli Li is currently a machine learning lab head in the Data Analytics Department at the Institute for InfoComm Research, Agency for Science, Technology and Research, Singapore. He also holds an appointment of adjunct assistant professor at the School of Computer Engineering, Nanyang Technological University. His research interests include data mining, machine learning, and bioinformatics. He has been serving as a member of technical program committees in the leading data mining and machine learning conferences such as KDD, ICDM, SDM, PKDD/ECML, PAKDD, WWW, ACML, AAAI, and CIKM. He is the coeditor-in-chief of the International Journal of Knowledge Discovery in Bioinformatics. He received a number of best paper awards in international conferences (DASFAA 2011, GIW 2005) as well as global benchmarking awards (DREAM 2007, Opportunity Activity Recognition Challenge 2011).

Bin Wu received the BS degree from the Beijing University of Posts and Telecommunications in 1991, and the MS and PhD degrees from the ICT of Chinese Academic of Sciences in 1998 and 2002, respectively. He joined the Beijing University of Posts and Telecommunications as a lecturer in 2002, and is a professor at present. His research interests include data mining, complex network, and cloud computing. He has published more than 100 papers in refereed journals and conferences. He is a member of the IEEE.