IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 8, 2020, accepted July 22, 2020, date of publication July 29, 2020, date of current version August 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012670

Research on Service Discovery Methods Based on Knowledge Graph

LI GUODONG, QIU ZHANG, YONGKAI DING, AND WANG ZHE
School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, China

Corresponding author: Li Guodong (lgd@ncepu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC0831404.

ABSTRACT Service discovery is one of the main goals of a service-oriented architecture, helping to discover complex applications. Knowledge graph is a cross-cutting research hotspot in the fields of computer, knowledge engineering and information science. This paper makes in-depth research on the key issues of knowledge graph matching for service-oriented discovery. Firstly, construct a knowledge graph for service discovery, and then design a template to match the knowledge graph. By designing a matching template, the service discovery mechanism is transformed into a knowledge graph knowledge query. Finally, the validity of the knowledge graph matching method proposed in this paper is verified by experiments.

INDEX TERMS Service discovery, knowledge graph, template matching.

I. INTRODUCTION

Driven by the rapid development of the Internet, Web services have shown an explosive growth trend in both quantity and type [1]. With the continuous update of the service network environment, network services also change dynamically. Therefore, how to correctly and effectively find useful services has become an urgent issue.

Web services are self-describing, self-contained, modular applications that can be accessed through the Internet. They provide an effective way to solve business process execution and application integration [2]. In the early days, most services described by WSDL were matched based on keywords, which made them achieve matching at the syntax level only. Many semantically similar keywords do not match well. Later, Tim Berners-Lee proposed the concept of semantic web in 2000, which laid the foundation for semantic web service discovery. Many web service description languages such as OWL-S [3] and WSMO [4] have been proposed to describe web services using ontology and semantic web technologies.

Knowledge graph can describe concepts, entities, and the relationship between them and represent them in a structured form, which helps people to organize, manage, and understand the vast amount of information on the internet [5]. The development stage of the knowledge graph has gone through five processes including the pre-knowledge engineering period, the expert system period, the World Wide Web period, the group intelligence period, and the knowledge graph period [6]. The current knowledge graph has become a powerful asset for semantic search, big data analysis, intelligent recommendation and data integration. Therefore, combining graphs with service discovery is also in line with the development trend.

The main research content of this paper is the matching problem of knowledge graph for service-oriented discovery. This paper is divided into five sections, the specific content and organizational structure are as follows:

(1) The section I is the introduction. This section mainly introduces the background, research purpose and significance of knowledge graph for service discovery.

(2) The section II is the related work. This section mainly introduces the research status of Web service discovery.

(3)The section III is the construction of knowledge graph. This section mainly introduces the construction of knowledge graph in detail, which contains knowledge extraction, knowledge processing and service link.

(4) The section IV is the construction of knowledge graph matching template for polygon service discovery. This section mainly introduces matching template for service discovery based on knowledge graph.

(5) The section V is the experiment. We verified the effectiveness of knowledge graph for service discovery by experiments, which contains user's intent classification validity and accuracy of entity recognition.

(6) The section VI is the conclusion. In this section, we summarize the work of this paper and put forward prospects for further research in the future.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhangbing Zhou.

II. RELATED WORK

Web service discovery has roughly gone through three stages: keyword-based service matching methods, syntax-based service matching methods, and semantic-based service matching methods. Due to the lack of semantic description in the first two methods, the accuracy of its discovery is low, and it cannot meet the needs of web service development. In order to overcome the disadvantages of traditional web service discovery methods, semantic-based web service discovery methods have emerged as the times require.

Semantic Web services discovery is a hot topic in the field of service science. Most semantic matching is through matching of service operations and corresponding service input and output parameters.

Sangers [7] *et al.* created a service context. This context is composed of keywords extracted from the service description. Then it integrates natural language processing technology with service context into the keyword-based discovery process. Elgazzar *et al.* [8] proposed a method for discovering that the service exactly matches the subtasks in the mobile environment. Exact matching means that all subtask inputs and outputs exactly match the service input and output parameters. A common feature of the above service discovery methods is that they treat semantics as a set of terms or annotated concepts and ignore the relationships in the ontology. As a result, they are not enough to effectively discover services, especially when logic-based reasoning is needed.

Some researchers have explored ways to make up for above limitation. Georgios and Nick use an ontology-based framework to retrieve Web services. The framework contains reasoning based on containment relationships and structure-based cases [9]. Under certain constraints, these service discovery methods perform better than keyword-based methods. However, they are time consuming due to the complicated way of reasoning. Paliwal *et al.* used the semantic classification of Web services to mine the associations of terms to solve the big data challenge of service discovery in large service pools [10]. Obviously, if services are pre-grouped, service discovery will be faster. Therefore, Kim *et al.* [11] proposed a cluster-based method to discover services in the MANET environment, and clustered similar services together, effectively improving the efficiency of service discovery.

Although many efforts have been made to optimize semantic Web service discovery, there is still considerable room to improve accuracy and efficiency.

Knowledge graphs show entities, events and their relationships in a graphical way. The storage and query of knowledge graphs mainly study how to design an effective storage mode to support the effective management of large-scale graph data and achieve efficient query of knowledge in knowledge graphs. This paper combines knowledge graph and service discovery technology to transform traditional service discovery methods into knowledge graph matching technology.

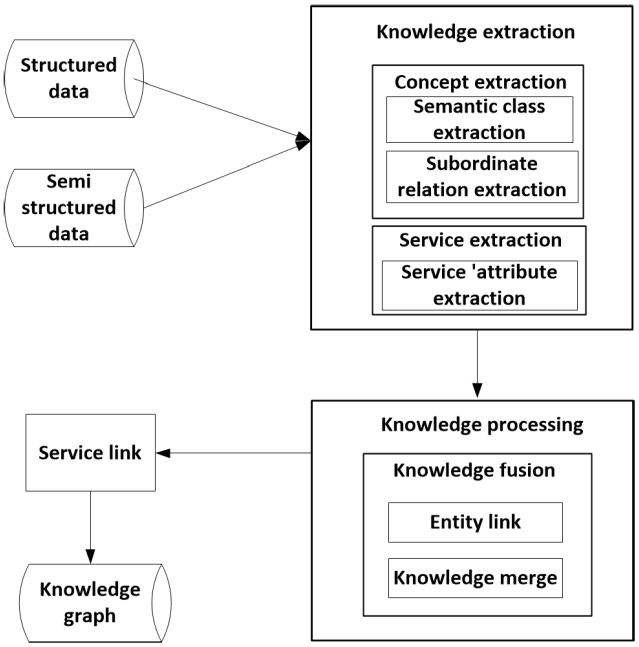

FIGURE 1. The model for building a knowledge graph for service-oriented discovery.

III. CONSTRUCTION OF KNOWLEDGE GRAPH

The knowledge graph is usually divided into two levels: the pattern layer and the data layer. The pattern layer is usually built on top of the data layer and stores abstracted knowledge, rules, and patterns. The data layer stores real data. Figure 1 shows a model for building a knowledge graph for service-oriented discovery.

As can be seen from the figure, the knowledge elements are first extracted from structured and semi-structured data through a series of automated or semi-automated technical means, and then the knowledge is processed. After knowledge fusion and knowledge processing, it is stored in the knowledge graph. Finally, the pattern layer and the data layer are linked through service links to form a large knowledge graph.

The construction of knowledge graph is a process of iterative construction. It can be seen from the figure that each round of iterative construction includes three stages, as follows:

A. KNOWLEDGE EXTRACTION

Extract entities, attributes and their relationships from web pages or data sources with different Internet structures, and transform them into structured data, such as RDF triples. It can be seen from the figure that the core of the knowledge graph of service-oriented discovery is knowledge extraction, which is divided into two parts: concept extraction and service extraction.

1) KNOWLEDGE PROCESSING

Including knowledge fusion and knowledge processing. Knowledge fusion mainly deals with the acquired knowledge

to eliminate the contradiction and ambiguity between knowledge. For example, some entities may have different meanings in different contexts, and a certain relationship can correspond to multiple entities, etc. Knowledge processing is a new data after audit and fusion. Only after quality assessment can data be added to the knowledge graph.

2) SERVICE LINK

The corresponding service data should be added to the knowledge graph after the construction of the upper model. Link services to a knowledge graph based on their capabilities.

The construction of knowledge graph is described in detail below.

B. KNOWLEDGE EXTRACTION

The knowledge extraction in this paper includes concept extraction and service extraction.

Concept extraction is divided into two parts: semantic class extraction and hyponymy extraction.

Semantic class can be understood as the concept of ''class'' in object-oriented or entity. We get entities by crawling Baidu Encyclopedia and other websites. For example, the ''Supreme Court'', ''high court'', ''intermediate court'' and ''basic court'' on the judicial website are all legal related entities.

Hyponymy extraction is to extract the hyponymy of a class or word from a document and generate data pairs. Because there are parallel or inclusive relationships between entities. In order to link the extracted knowledge into a large knowledge network, we also need to extract the relationship between entities, which is the bridge connecting entities. For example: high court includes Heilongjiang High People's court.

By analyzing the classification of Wikipedia and the open classification of Baidu Encyclopedia, a series of semantic classes and upper and lower relations are obtained. As shown in Figure 2, take the judicial field as an example.

Service extraction includes service's attribute value extraction and relationship extraction. The task of attribute value extraction is to construct attribute list for each service, so as to form a complete service. Property contains the service name, access address, input and output, and function of the service. Taking ''predict legal charges according to crime details'' service as an example, we define the storage mode of service data, as shown in TABLE 1.

As can be seen from TABLE 1, service is determined by five attributes: service name, service input, service output, service function and service address. As long as one of the five attributes is different, it is a different service.

C. KNOWLEDGE PROCESSING

Through knowledge extraction, we get entity, relationship and attribute information from the original structured and semi-structured data. But the data is still disordered, which needs to be processed to form structured data in the knowledge base.

Knowledge fusion is the fusion of knowledge extracted from different sources. Knowledge fusion includes entity

FIGURE 2. A series of semantic classes and upper and lower relations in judicial field.

TABLE 1. Service data storage mode.

Service properties	Parameter instance
service name	crimePrediction
service input	details of the crime
service output	charge
service function	predict crimes based on crime details
service address	"http://localhost:8089/domains/1.1/travel/city_hotel_service.owls"

linking and knowledge merging. Entity linking refers to the operation of linking the entity object to the corresponding correct entity object in the knowledge graph. For example: ''at the Apple conference in jobs theatre in 2019, the latest iPhone 11 was launched.''. A complete entity link includes the following studies:

(1) To recognize entity references in natural language is entity recognition. The so-called entity referent is the object that we want to link to the knowledge graph. For example (jobs theater, apple, iPhone 11), etc.

(2) For each entity reference, the abstract concepts that it may point to in the knowledge graph are identified. In the above example apple may point to abstract concepts such as fruit to Apple, company to Apple, etc.

(3) Determine the correct category of entity references according to their context. For example, according to the context of Apple's jobs theater, mobile phone, iPhone 11, apple belongs to the category of company, not the movie or fruit.

D. SERVICE LINK

Service link is the proper link of the processed service data and abstract concept data. Link is to find the most suitable path node in the knowledge graph according to the function description of service, and put it under the concept of corresponding path. Syntactic analysis provides us with a solution. Syntactic analysis tree is to generate a tree according to

TABLE 2. The parsing tree annotation.

Parsing tree annotation	Meaning
ROOT	natural language to deal with
NP	noun phrase
NN	singular form of common nouns
LCP	location phrase
NT	temporal noun
VV	verb
VP	verb phrase
PU	punctuation

TABLE 3. The service properties of findCase.

Service properties	Parameter instance
service name	findCase
service input	geographical location, procuratorate
service output	criminal case
service function	finding criminal cases of Chengdu people's Procuratorate in the past month
service address	"http://localhost:8089/domains/1.1/legal/find_case.owls"

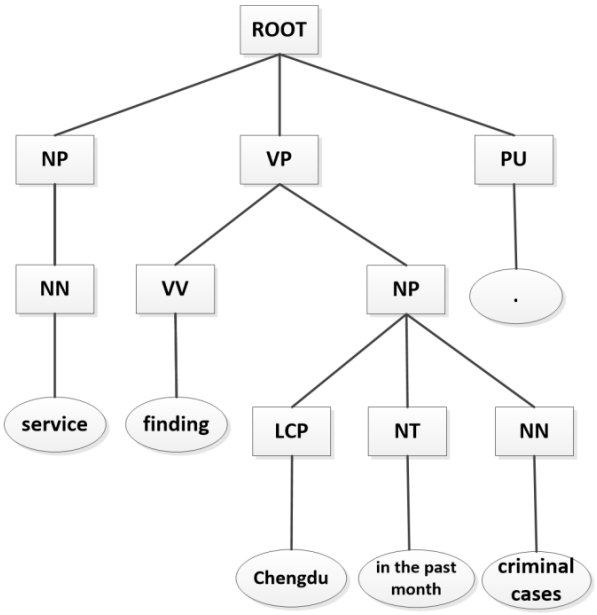

FIGURE 3. The generated syntax analysis tree.

the syntactic logic of a sentence. Syntactic analysis includes syntactic structure analysis and dependency analysis. This paper only uses syntactic structure analysis. TABLE 2 shows the commonly used annotation set of parsing trees.

Take the service of finding criminal cases of Chengdu people's Procuratorate in the past month as an example, the structure of the service is shown in the TABLE 3.

According to the service's structure, the generated syntax analysis tree is shown in Figure 3.

Through the syntactic analysis tree, we can find the structure and semantics that are closest to the pattern layer of knowledge base. The node to be linked by the service must

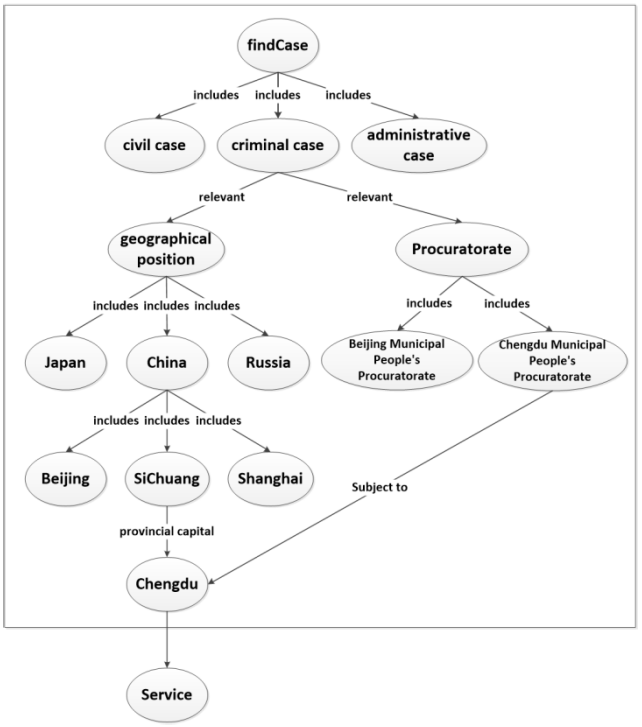

FIGURE 4. The example of knowledge graph based on syntactic analysis tree.

be on the longest matching path. As shown in the solid wireframe in Figure 4, it is a part of the knowledge graph, and the location of the service linked to the map is the best location.

IV. CONSTRUCTION OF KNOWLEDGE GRAPH MATCHING TEMPLATE FOR POLYGON SERVICE DISCOVERY

The construction of knowledge graph query template for service discovery focuses on the problem of transforming natural language questions into knowledge query in combination with knowledge graph. Figure 5 is the architecture of knowledge graph query template for service discovery. The rounded rectangular box pointed by the arrow in the figure is the four steps of service discovery, namely: user intention classification, service entity recognition, query template construction and similarity calculation.

It can be seen from the Figure 5 that after the user inputs a natural language problem and processes it through different modules, the relevant results of the problem can be found in the knowledge graph. The construction of knowledge graph is described in detail below.

A. CLASSIFY THE USER'S INTENT

The classification of user's intent is to understand the user's problem and classify the user's requested intent. For example: "Information on luxury hotels in Paris, France" is a tourism issue, and "Number of cases in Henan Province in March 2019" is a legal issue.

There are two main difficulties in the classification of user's intent. First of all, the user input is not standardized, and the input methods are various. The second is that the query words used by users show multiple intentions.

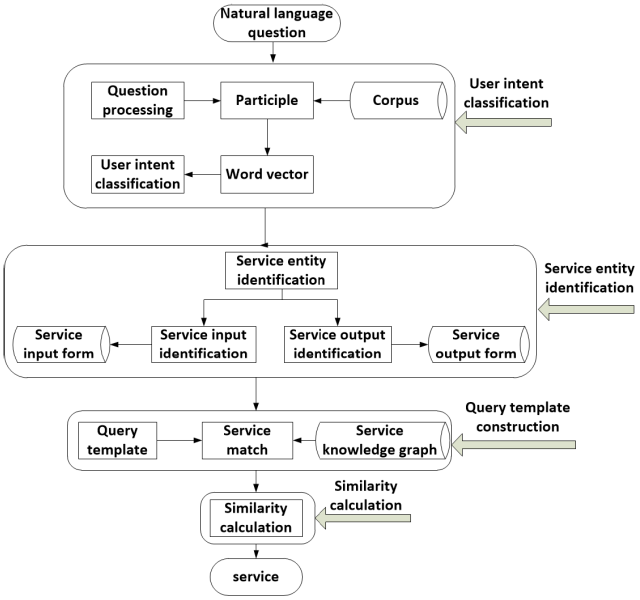

FIGURE 5. The architecture of knowledge graph query template for service discovery.

For example, when users search for ''apple'', do they mean mobile phones or fruits? Therefore, we adopt the following process to solve the above problems.

1) PREPARATION STAGE

Before the classification of user's intent, we need to process the natural language input by the user, mainly word segmentation and removing stop words. Since Chinese is not based on words as in English and there are no spaces between words as a distinction, the analysis of Chinese natural language must consider the overall semantics.

We use jieba [12] word segmentation tool to segment natural language. For example: after word segmentation, ''number of cases in Henan Province in March 2019'' is converted into ''2019/ March / Henan Province / cases / number''.

After word segmentation, natural language needs to be transformed into word vector, which is mainly completed by training wiki corpus with word2vec tool. The final word vector is 200 dimensions. Considering these five words, we use skip-gram method to generate the vector.

2) TRAINING STAGE

In this paper, we use naive Bayes algorithm to classify user's intent. By calculating the frequency of each category in the training sample, and dividing the conditional probability estimation of each category for each feature attribute, finally recording the results. In this stage, the input is the feature attribute and labels of training set, and the output is the trained classification model.

3) TEST STAGE

The task of this stage is to use the naive Bayes classification model generated in stage 2 to classify the samples to be tested. The input is the classification model and test samples, and the output is the mapping relationship between the items to be classified and the categories.

Therefore we can use the trained model to classify user's intent.

TABLE 4. The service input and output recognition label set.

Entity category	Examples	label
Time	2019, July, one month	0
Geographical location	Qinghai Province, New York	1
Inspection authority	Supreme people's procuratorate	2
Court	Beijing First Intermediate People's Court	3
Criminal	Fraud, Corruption	4
Prison term	One month in prison	5
Penalty	deprived of political rights	6
Defendant	Mr. Zhang	7
Plaintiff	Mr. Hang	8
Case category	corruption case	9
Document number	No. 290	10

B. SERVICE ENTITY RECOGNITION

After classification, the user's intent can be roughly determined, so the precise query can be performed in a smaller range. Service entity recognition plays a key role in query. Entity is an important language unit that carries information in text. The semantics of a text can be expressed as the association and interaction between entities. Entity recognition is to recognize the entity with specific meaning in the text.

The section III introduces the attributes of service, including name, input, output, function and address. Therefore, service entity recognition includes service input entity recognition and service output entity recognition. Taking justice as an example, it includes input table and output table. TABLE 4 shows examples of service input and output recognition labels.

With the development of deep learning, more and more deep learning methods are being used for entity recognition [13]. Graph LSTM is developed on the basis of LSTM [14]. Graph LSTM performs entity recognition through ''forget gate'', ''input gate'', ''output gate'' and memory unit. Unlike LSTM, the logical structure of Graph LSTM is graph-like. At a certain moment, the current node can learn the information of all neighboring nodes connected to the node and transfer it to the next node. The entity recognition problem can also be expressed as an inference problem on the local neighborhood of the graph [15]. Graph LSTM makes it possible to learn features directly from examples by bypassing the steps of creating and adjusting inference models [16].

This paper uses Graph LSTM neural network architecture based on LSTM for entity recognition. Graph LSTM

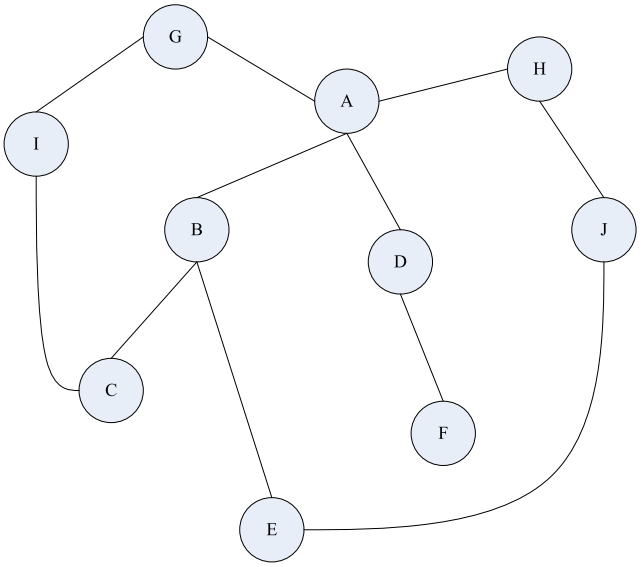

FIGURE 6. The abstract structure of Graph LSTM.

is developed on the basis of LSTM. Graph LSTM uses ''forgetting gate'', ''input gate'', ''output gate'' and memory unit for entity recognition. Unlike LSTM, graph LSTM has a graphical logical structure. At a certain time, the current node can learn the information of all neighboring nodes, and pass it to the next node. The problem of entity recognition can also be expressed as an inferential problem on the local neighborhood of graph. Graph LSTM makes it possible to learn features directly from examples, instead of creating and adjusting reasoning models.

Given a graph structure G(V, E), where V is a set of vertices and E is a set of edges. Suppose that each vertex v is labeled with the eigenvector g (v), and the edge represents the association between the two nodes. L (v) denotes the category label of vertex v. Figure 6 shows the abstract structure of Graph LSTM, where each node stores feature vectors related to sentences. The neighborhood of each node contains information that allows inference and prediction. In this paper, the prediction of nodes is generated according to the characteristics of its neighboring nodes.

In order to identify the category label of entity V, the graph can be extended to a tree with V as its root and D as its neighborhood radius. We select node A as the root node and select the neighborhood radius of 2 to expand the graph structure into a tree. Because the neighborhood radius is set to 2, the two paths A-B-C-I and A-B-E-J are not shown in Figure 7. As shown in Figure 7, the depth of node A is zero, traverses from bottom to top, and calculates the label of each parent node according to the characteristics of its child nodes. Finally, the category label of root a is generated, and the model output is compared with manually marked categories.

The process of entity recognition consists of two steps: forward propagation and backpropagation. Forward propagation follows the tree from the bottom to the top. Figure 8 shows a forward propagation process. As shown in the figure, taking node I as an example, it represents the word vector of the current node. Starting from node I, the useful

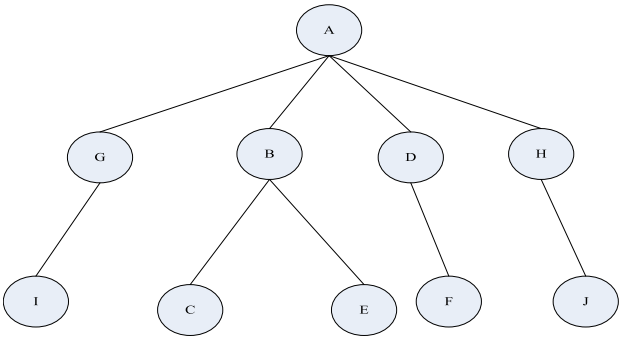

FIGURE 7. The extended tree structure of Graph LSTM.

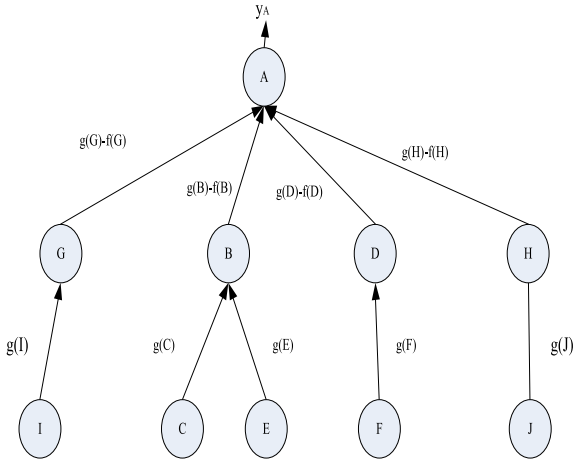

FIGURE 8. The forward propagation process of Graph LSTM.

information learned is passed up to node G. It represents the integration of the useful knowledge left after passing through the forgetting gate and the input gate. According to the above process, it is passed layer by layer until the root node A. finally, the class label of node A is recognized.

After the forward propagation, the state of each node needs to be recorded to facilitate the back propagation. We also need to find the parameters to optimize model. In the early training of the model, the accuracy is generally not high, and the output results are often different from the expected results.

Therefore, it's need to calculate the error term value of each neuron to construct the loss function for model optimization. Graph LSTM back propagation propagates from the root node to the leaf node, and uses the gradient descent method to update the loss function of each node. According to the gradient guidance of loss function L, the network weight parameters are updated. We propagate step by step as follows:

$$\delta(\mathrm{h_t}) = \frac{\partial L}{\partial h_t}$$
$$\delta(\mathrm{c_t}) = \frac{\partial L}{\partial c_t}$$

C_t is like a conveyor belt, can control the transmission of information to the next node. Each node has a hidden state, which is recorded as h_t to store the state information of the previous node.

We select different nodes as root nodes to expand the neighborhood tree, which can fully learn the information

between words and carry out entity recognition. The neighbor node whose radius is 2 from the target node is selected to expand into tree, which not only saves the traversal time, but also improves the accuracy.

C. QUERY TEMPLATE CONSTRUCTION

Template is formed after fixing and standardizing the structural law of things, which reflects the standardization of structural form. Query template is constructed after user's intent classification and service entity recognition, which can dynamically build query template and make query template extensible. For example, ''search for traffic accident cases in April 2018 in Hunan Province'', a complete template generation process includes the following steps.

(1) Eliminate stop words and segmentation. As there is no stop word in ''search for the traffic accident case in April 2018 in Hunan Province'', the sentence is segmented into: ''find/Hunan Province/2018/April/of/traffic accident case/''.

(2) Analyze the user's query intent to determine problem category. Analyzing this sentence, we know that it belongs to case finding service.

(3) Identify the input and output of the service. The user's intent in ''Finding the Traffic Accidents in Hunan Province in April 2018'' is case finding, and the inputs for the service are ''Hunan Province'', ''2018 April'' and ''traffic accident'', besides the output of the service is ''case''.

(4) Replace the input and output of the identified services with their corresponding categories. For example: Hunan Province → <input: location>, April 2018 → <input: date>, traffic accident → <input: case>, <output: case>.

(5) Generate the query template. As follows:

[''MATCH (m:serviceItem) where m.input = '{location,date,case}' and m.output = '{case}' return m'']

Where ''serviceItem'' represents the category to which the user's query intention belongs, expressed by m. And ''m.output'' represents the output entity of the service under a category. Besides ''m.input'' represents the input entity of a service under a category. Finally ''return m'' represents the service that is finally returned.

As for ''search for the traffic accident case in April 2018 in Hunan Province'', through the above process we return a service named as findTrafficCase which meets user's intent, as shown in TABLE 5.

This section mainly introduces the construction process of knowledge graph query template for service-oriented discovery. Firstly, we classify user's service query intention and limit user's intention to a specific range. Then we identify the input and output entity. Finally, we can dynamically build query templates to query the service which meets user's intention in the knowledge graph.

V. EXPERIMENT

Earlier we introduced how to build a knowledge graph for service-oriented discovery and the generation of query templates based on knowledge graph. In this chapter, the effectiveness of service discovery is verified by experiments.

TABLE 5. The service properties of find TrafficCase.

Service properties	Parameter instance
service name	findTrafficCase
service input	geographical location, date
service output	case
service function	search for traffic accident cases at a certain time according to geographical location
service address	"http://localhost:8089/domains/1.1/travel/find_traffic_case.owls"

TABLE 6. The parameter setting of naïve Bayes algorithm.

Parameter	Value	Meaning
Alpha	1	prior smoothing factor
fit_prior	true	prior probability
class_prior	None	probability of categories

A. USER'S INTENT CLASSIFICATION VALIDITY

In this part we use Bayesian naive classification algorithm and Support Vector Machine algorithm to classify the user's intent in judicial query.

The corpus of this experiment was crawled from some websites such as China's referee's website, Baidu encyclopedia, Wikipedia and so on. Some of the data were from OWLS-TC4-PDDL's domain knowledge, totaling more than 30000. In order to obtain as much effective information as possible from the limited data and prevent over fitting, we test the accuracy of the algorithm by using ten folds cross validation. The data set is randomly divided into ten parts, nine of which are used as training sets and the rest as test sets. The average of the accuracy of the ten results is used to estimate the accuracy of the algorithm.

The experimental procedures are as follows:

(1) data preprocessing. Jieba Thesaurus is used for word segmentation.

(2) eliminate stop words, filter out irrelevant punctuation, mood auxiliary words and so on.

(3) use word2vec tool to convert the text into vectors which our model can handle with.

(4) divide the dataset into training set and test set. The classification model learns the features of the training set, and then uses the test set to test the performance of the classification model.

The naïve Bayes parameter settings are shown in TABLE 6.

The prior smoothing factor is set to 1 for adding Laplacian smoothing to naive Bayesian model. If a word does not appear in the training sample, the probability of the word will be zero, which is very unreasonable. Laplacian smoothing is to solve the problem of zero probability. In addition, the prior probability of each category is considered in the training process. If a priori probability is not considered, all the sample

TABLE 7. The parameter setting of SVM algorithm.

Parameter	Value	Meaning
C	10	penalty parameter
kernel	RBF	kernel function
gamma	auto	kernel function parameter

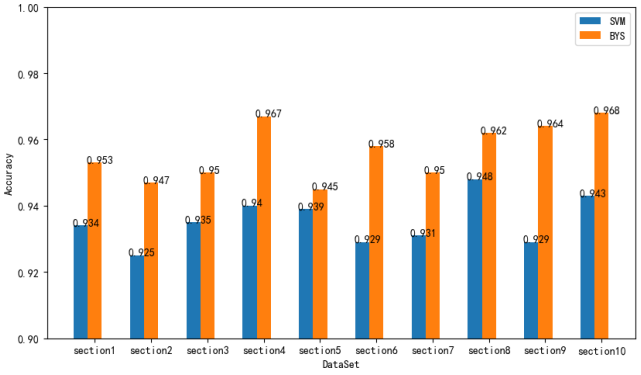

FIGURE 9. The classification of user's intent.

class outputs have the same class prior probability, which will affect the accuracy of classification. Although a priori probability is considered, the prior probability of each category is not specified, and the prior probability is obtained by automatic learning of the model.

The SVM parameter settings are shown in TABLE 7.

The penalty parameter represents the degree of punishment for classification errors. The larger the penalty parameter is, the smaller the classification error is not allowed. However, if the penalty parameter is too large, it may cause over fitting. kernel function is imported to solve the problem of linear indivisibility.

This paper verifies the performance of the naïve Bayesian and SVM algorithm. The experimental results are shown in Figure 9. The orange cylinder is the accuracy of the Bayesian algorithm, and the blue part is the accuracy of the SVM algorithm. The X-axis is the number of experiments and the y-axis is the classification accuracy. Section1-section10 is the ten experiments performed.

As shown in Figure 9, the average accuracy of SVM algorithm is 93.53%, and the average accuracy of naïve Bayesian algorithm is 95.64%. Therefore, the accuracy of Bayesian algorithm is higher than that of SVM algorithm.

B. ACCURACY OF ENTITY RECOGNITION

The validity of entity recognition using Graph LSTM is then verified. The experiments are compared from two aspects. On the one hand, it compares the differences in recognition accuracy, recall rate, and F value between LSTM and Graph LSTM. On the other hand, the efficiency of the proposed method is evaluated by comparing the execution time. Comparison of accuracy of entity recognition. As shown in Table 3.

TABLE 8. The result of service entity recognition.

	precision	recall	F-measure
LSTM	95.7	89.2	87.9
Graph LSTM	96.3	90.4	91.6

The data used in this experiment includes 17 kinds of legal terms, such as time, place, procuratorial organ, accusation and so on, which are crawled from judicial website, Baidu Encyclopedia and Wikipedia, with a total of more than 5000 entities. 70% of them are training set and 30% are test set.

The Graph LSTM execution process is as follows:

(1) the root node is randomly selected to construct the tree, the depth is set to 2, and then the propagation is executed along the tree;

(2) the output contains the final entity category. Update network parameters after each batch.

The process above was randomly executed 50 times.

In addition, we use SoftMax activation function and adaptive learning rate algorithm to improve Stochastic Gradient Descent(SGD) algorithm instead of manual adjustment of learning rate. The learning factor of SGD algorithm is set to 1.0, the attenuation parameter is set to 0.95, and the number of iterations is 2. LSTM algorithm parameter setting: set the output dimension to 128, batch_ The size is set to 30, the number of iterations is 20, and the dropout rate is 0.5.

The precision, recall rate and F-measure value were used to evaluate the performance of the model. The calculation formula is as follows:

$$\mathtt{precision}(A) = \frac{TP + \mathrm{TN}}{TP + TN + FN + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F\text{-}measure = \frac{2\mathrm{p} * \mathrm{r}}{p + r}$$

where TP: true positive, TN: true negative, FP: false positive, FN: false negative.

The experiment was compared from two aspects. On the one hand, the differences of recognition precision, recall rate and F-measure value between LSTM and graph LSTM are compared. On the other hand, the efficiency of the proposed method is evaluated by comparing the execution time.

The service entity recognition result is shown as follows.

The efficiency comparison between LSTM and Graph LSTM algorithms is shown in Figure 10.

The overall experimental results show that the service entity recognition method using graph LSTM proposed in this paper has high computational efficiency and accuracy.

This chapter mainly introduces the experiments involved in this paper. The first is user intention classification experiment. In this paper, naive Bayesian classification algorithm is used for classification, and SVM is used as the comparative experiment. The experimental results show that naive

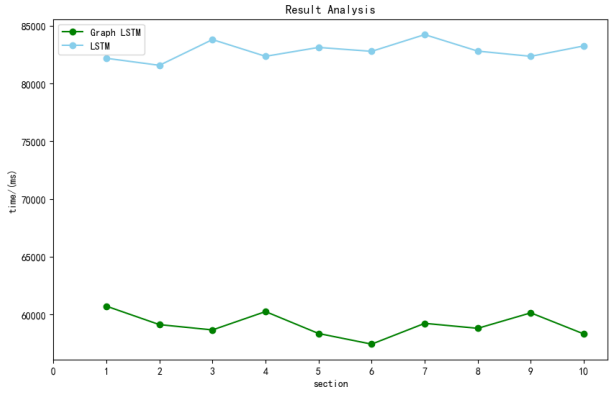

FIGURE 10. The efficiency comparison between entity recognition algorithms.

Bayesian algorithm can effectively classify the user's intention. And then, the efficiency and accuracy of graph LSTM and LSTM are compared to verify the effectiveness of the method.

VI. CONCLUSION

This paper makes in-depth research on the problem of service discovery based on knowledge graph matching. This article discusses how to construct a service knowledge graph, how to perform service discovery on the service knowledge graph, and designs a service knowledge graph construction method and a knowledge graph template matching method. The experiment proves that the scheme works well.

The contributions of this paper are as follows:

1) BUILDING A SERVICE-BASED KNOWLEDGE GRAPH

Service entities are recognized from structured and semi-structured data to form the concept layer of knowledge graph. Then, by analyzing the characteristics of the service, we extract the input and output parameters, the function and address of the service. Finally, The service is linked to the service entities to form a complete knowledge graph.

2) CARRYING OUT SERVICE DISCOVERY THROUGH THE CONSTRUCTED KNOWLEDGE GRAPH

The knowledge graph is stored in the graph database neo4j. Neo4j supports cypher, a structured database query language. Therefore, the process of service discovery in this paper is to transform the user's service request (i.e. natural language) into query language of graph database and query in knowledge graph. This paper constructs a query template, which can dynamically and real-time build query language query graph database.

3) DESIGN EXPERIMENTS TO VERIFY THE USABILITY AND EFFECTIVENESS OF THIS WORK

First, we verify the accuracy of user intention classification. Experiments show that the accuracy of naive Bayes algorithm is higher than that of SVM algorithm. Then we verify the accuracy of entity recognition. Experiments show that graph LSTM method is better than LSTM in recognition in accuracy and efficiency.

The future work is to expand the application range of the method proposed in the paper. The knowledge graph for service-oriented discovery constructed in this article is mainly for legal services, but there are a variety of services on the Internet, which are spread across all walks of life. Such as tourism services, medical services, economic services and so on. These services are far from legal services. The input, output and functions are also different. How to turn the domain service knowledge graph into a general service knowledge graph is a very important research direction.

REFERENCES

[1] D. Zhu, "Research of semantic Web services discovery based on clustering and bipartite graph," *Comput. Eng.*, vol. 42, no. 2, pp. 157–163, 2016.
[2] L. Zhizhong, W. Huaimin, and G. Yanling, "A semantic service discovery model based on two-tier P2P architecture," *Comput. Eng. Sci.*, vol. 50, pp. 135–142, 2007.
[3] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara, "OWL-S: Semantic markup for Web services," *W3C Member Submission*, vol. 22, p. 4, Nov. 2004.
[4] J. Domingue, D. Roman, and M. Stollberg, "Web service modeling ontology (WSMO)—An ontology for semantic Web services," *Inf. Syst. e-Bus. Manage.*, vol. 3, no. 2, pp. 175–199, 2005.
[5] R. Lijuan, L. Jun, and G. Wei, "Multi-source knowledge embedding research of knowledge graph," in *Proc. IEEE 3rd Int. Conf. Circuits, Syst. Devices (ICCSD)*, Aug. 2019, pp. 163–166.
[6] *Language and Knowledge Computing Special Committee of Chinese Information Society of China. Knowledge Graph Development Report*, Lang. Knowl. Comput. Special Committee of Chin. Inf. Soc. China, Beijing, China, 2018.
[7] J. Sangers, F. Frasincar, F. Hogenboom, and V. Chepegin, "Semantic Web service discovery using natural language processing techniques," *Expert Syst. Appl.*, vol. 40, no. 11, pp. 4660–4671, Sep. 2013.
[8] K. Elgazzar, H. S. Hassanein, and P. Martin, "DaaS: Cloud-based mobile Web service discovery," *Pervas. Mobile Comput.*, vol. 13, pp. 67–84, Aug. 2014.
[9] G. Meditskos and N. Bassiliades, "Structural and role-oriented Web service discovery with taxonomies in OWL-S," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 2, pp. 278–290, Feb. 2010.
[10] A. V. Paliwal, B. Shafiq, J. Vaidya, H. Xiong, and N. Adam, "Semantics-based automated service discovery," *IEEE Trans. Services Comput.*, vol. 5, no. 2, pp. 260–275, Apr. 2012.
[11] Y.-S. Kim, Y.-S. Shim, and K.-H. Lee, "A cluster-based Web service discovery in MANET environments," *Mobile Inf. Syst.*, vol. 7, no. 4, pp. 299–315, 2011.
[12] J. J. Sun. *Jieba Chinese Word Segmentation Tool*. Accessed: Jul. 1, 2019. [Online]. Available: https://github.com/fxsjy/jieba
[13] N. Mahanta, S. Dhar, and S. Roy, "Entity recognition in assamese text," in *Proc. Int. Conf. Commun. Electron. Syst. (ICCES)*, Oct. 2016, pp. 1–5.
[14] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan, "Semantic object parsing with graph LSTM," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 125–143.
[15] R. Agrawal, L. de Alfaro, and V. Polychronopoulos, "Learning from graph neighborhoods using LSTMs," in *Proc. Workshops 31st AAAI Conf. Artif. Intell.*, 2017, pp. 377–384.
[16] G. Li, Z. Wang, and Y. Ma, "Combining domain knowledge extraction with graph long short-term memory for learning classification of Chinese legal documents," *IEEE Access*, vol. 7, pp. 139616–139627, 2019.

LI GUODONG received the Ph.D. degree in control theory and control engineering from North China Electric Power University, China, in 2017. He is currently an Associate Professor with the School of Control and Computer Engineering, North China Electric Power University. His research interests include computer network application and service computing.

QIU ZHANG is currently pursuing the master's degree in software engineering with North China Electric Power University, Beijing, China. His research interests include web service discovery and machine learning.

YONGKAI DING is currently pursuing the master's degree in computer application technology with North China Electric Power University, Beijing, China. His research interest includes web service discovery.

WANG ZHE is currently pursuing the master's degree in computer application technology with North China Electric Power University, Beijing, China. Her research interests include knowledge engineering, natural language processing, and big data analysis and processing.