Computer and Information Science; Vol. 14, No. 2; 2021
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Research on Entity Label Value Assignment Method in Knowledge Graph

Linqing Yang[1], Bo Liu[1], Youpei Huang[1], & Xiaozhuo Li[1]

[1] School of Information Science and Technology, Jinan University, Guangzhou, China

Correspondence: Linqing Yang, School of Information Science and Technology, Jinan University, Guangzhou, China.

Received: February 24, 2021 Accepted: March 26, 2021 Online Published: April 13, 2021

doi:10.5539/cis.v14n2p63 URL: https://doi.org/10.5539/cis.v14n2p63

Abstract

The lack of entity label values is one of the problems faced by the application of Knowledge Graph. The method of automatically assigning entity label values still has shortcomings, such as costing more resources during training, leading to inaccurate label value assignment because of lacking entity semantics. In this paper, oriented to domain-specific Knowledge Graph, based on the situation that the initial entity label values of all triples are completely unknown, an Entity Label Value Assignment Method (ELVAM) based on external resources and entropy is proposed. ELVAM first constructs a Relationship Triples Cluster according to the relationship type, and randomly extracts the triples data from each cluster to form a Relationship Triples Subset; then collects the extended semantic text of the entities in the subset from the external resources to obtain nouns. Information Entropy and Conditional Entropy of the nouns are calculated through Ontology Category Hierarchy Graph, so as to obtain the entity label value with moderate granularity. Finally, the Label Triples Pattern of each Relationship Triples Cluster is summarized, and the corresponding entity is assigned the label value according to the pattern. The experimental results verify the effectiveness of ELVAM in assigning entity label values in Knowledge Graph.

Keywords: knowledge graph, entity label value, assignment, external resources, entropy

1. Introduction

Computers have been facing the dilemma of not being able to obtain the semantic information of the text. In order to allow the machine to understand the meaning behind the text, Knowledge Graph came into being. "Knowledge graph is a knowledge base, which aims to enhance search engine’s results by collecting information from various sources" ("Knowledge Graph"). With the development of computer science and technology, the concept of Knowledge Graph is also expanding. (Q. Liu, Li, Duan, Y. Liu, & Qin, 2016) extended the definition of Knowledge Graph and stated that "Knowledge Graph is a structured semantic knowledge base, which is used to describe concepts and their relationships in the physical world in symbolic form. Knowledge Graph consists of the entity-relationship-entity triples as well as the entity-property-property value triples. The entities are connected to each other through relationships, forming a networked knowledge structure". Among them, entity refers to something that is distinguishable and independent, such as a person, a city, a plant ("Entity"); relationship reflects the relationship between the entity and the entity, which is a kind of mapping; property is the inherent characteristics and the supplementary description of the entity.

The lack of data is a serious problem faced by the application of Knowledge Graph. Especially in many established Knowledge Graph, not only entities or relationships are missing, but also entity-property-property value triples are usually lacking. Therefore, some Knowledge Graph needs to be complemented or extended to complete the application requirement function.

The entity type (i.e., label) is one of the properties of the entity. In recent years, some scholars have proposed the methods of automatically assigning entity types to triples in Knowledge Graph (Moon, Jones, & Samatova, 2017; Neelakantan & Chang, 2015; Paulheim & Bizer, 2013; Nuzzolese, Gangemi, Presutti, Draicchio, Musetti, & Ciancarini, 2013). Given Knowledge Graph, where there are only entity-relationship-entity triples and no or only some entity-property-property value triples, these methods can automatically assign types to each entity in the triples. For example, for the triples "(Kismet, directed_by, William Dieterle)" whose entity types are unknown, the entity type assignment methods mentioned above will assign the type "Film" to the entity "Kismet" and the

type "Director" to the entity "William Dieterle".

In (Moon et al., 2017) and (Neelakantan & Chang, 2015), the authors propose to use an embedding model to infer the entity label values. Firstly, they use a negative sampling approach for data collection. That is, the entity or the label value of the triples is replaced with the entity or the label value of another triples. Secondly, they divide entity-relationship-entity triples and entity-label-label value triples into training set and test set. Thirdly, they train them through the designed loss function. It can be seen that using the embedding model to assign the entity label values not only takes a long time to train, but also costs the running space.

In the context of statistical distribution, Paulheim and Bizer (2013) propose a mechanism named SDType (Paulheim & Bizer, 2013), which is based on linking and weighted voting. SDType heuristically calculates the confidence of entity-type pairs from RDF knowledge base to reason about the missing types of entities. In the first step, subject-predicate-object triples and known entity-type pairs are treated as input. In the second step, based on every linked predicate, the number of triples for all entities is counted. In addition, the percentage of entities that are of that type is also counted. In the third step, the weight of each predicate as well as the conditional probability of each predicate-type pairs is calculated. In the final step, the confidence score for new entity-type pairs is computed to reason about the missing entity-type pairs.

Nuzzolese et al. (Nuzzolese et al., 2013) develop a model named Tipalo to automatically type the entities in DBpedia: First of all, Tipalo extracts the definition of a DBpedia entity from its corresponding Wikipedia page abstract, and relies on a set of heuristics based on lexico-syntactic patterns to obtain the shortest text including information about the entity type. Then, the machine parses the natural language definition of the entity and generates its OWL representation. Lastly, the type selector is designed to generate triples related to the entity and its type. The authors also map the types obtained from the type selector to other ontology categories. However, after mapping, precision and recall are not better than those when simply using the type selector.

The entity type prediction method based on the embedding model requires the training process, which has possess of high time complexity and high space complexity. SDType needs to know most of the entity-type pairs in advance, which is not applicable when all entity types are unknown. The heuristic templates proposed by Tipalo lack flexibility. There may exist the problem that RDFS infers some wrong types in Tipalo.

In domain-specific Knowledge Graph, if the label pattern of entity-relationship-entity triples can be found, the label values can be assigned to all entities. For example, if the Label Triples Pattern "(Film, directed_by, Director)" is found, the subject "Kismet" of the triples "(Kismet, directed_by, William Dieterle)" will be assigned the label value "Film" and the object "William Dieterle" will be assigned the label value "Director". Therefore, aiming at the absence of entity label values in domain-specific Knowledge Graph (i.e., there is no label value in Knowledge Graph), Entity Label Value Assignment Method based on external resources and entropy (ELVAM) is proposed in this paper. In ELVAM, the Label Triples Patterns are discovered first, and the label values are assigned to all entities next.

The research background of ELVAM proposed in this paper is not completely consistent with the embedding models and SDType model. Because these models require to know the corresponding types of some entities previously. The research background of ELVAM and Tipalo is similar, which means that the entity label values of all triples are completely unknown in Knowledge Graph. Compared with existing work, ELVAM has improved the time efficiency and the accuracy of entity label value assignment. The main contributions of this article are as follows:

1) For domain-specific Knowledge Graph, where the entity label values are completely unknown, ELVAM first determines the label values of the extracted entities, and then determines the label values of all entities in Knowledge Graph. We use a small amount of triples data to solve the problems of long training time and low efficiency.

2) ELVAM uses the crawler framework Scrapy ("Scrapy Tutorial") to crawl and F1-measure (Zheng, Cheng, Yu, Zou, & Zhao, 2019) to filter URL external resources for each Relationship Triples Subset, obtaining the text description of the entities. Based on this situation, we solve the problem of the lack of entity semantics.

3) ELVAM utilizes the synonym dictionary WordNet (Miller, 1998) and Ontology Category Hierarchy Graph ("Ontology Classes"), and calculates Information Entropy and Conditional Entropy (Shannon, 1948) to obtain entity label values with moderate granularity.

2. Related Definition

Definition 1 (Knowledge Graph *KG*) Knowledge Graph *KG* mentioned in this article is represented as *KG* = (*E*,

R, *P*, *PV*), where *E* is the set of entities, *R* is the set of relationships, *P* is the set of entity properties, *PV* is the set of entity property values. Supposing that label is noted as *l*, label value is noted as *lv*, label value set is noted as *LV*, then $l \in P$, $lv \in LV$, $LV \subseteq PV$.

Definition 2 (Triples *t*) Triples *t* is the basic unit of Knowledge Graph *KG*, composed of three elements: subject *s*, relationship *r*, and object *o*, namely $t = (s, r, o)$, $s \in E$, $r \in R$, $o \in E$.

Definition 3 (Relationship Triples Cluster *RTC*) Relationship Triples Cluster *RTC* is a cluster consisted of all triples of the same relationship type in Knowledge Graph *KG*.

Definition 4 (Relationship Triples Subset *RTS*) Relationship Triples Subset *RTS* is the subset of the Relationship Triples Cluster *RTC*, namely $RTS \subseteq RTC$.

Definition 5 (Entity Role *role*) Entity Roles are divided into two categories: subject role and object role. In the triples $t = (s, r, o)$, the role of *s* is the subject role, and the role of *o* is the object role.

Definition 6 (Label Triples Pattern *LTP*) The triples in the form like (lv_s, *r*, lv_o) is defined as LTP_r of the relationship *r*, where lv_s, $lv_o \in LV$.

Problem Definition Given the triples set *T* of domain-specific Knowledge Graph *KG*, as well as the initial label value set is empty, (i.e., $LV = \varnothing$), the label value lv_s and lv_o for the subject *s* and the object *o* respectively will be obtained after ELVAM, where lv_s, $lv_o \in LV$.

3. Entity Label Value Assignment Method Based on External Resources and Entropy

3.1 Overall Framework

The overall framework of ELVAM, mainly including four modules, is shown in Figure 1.

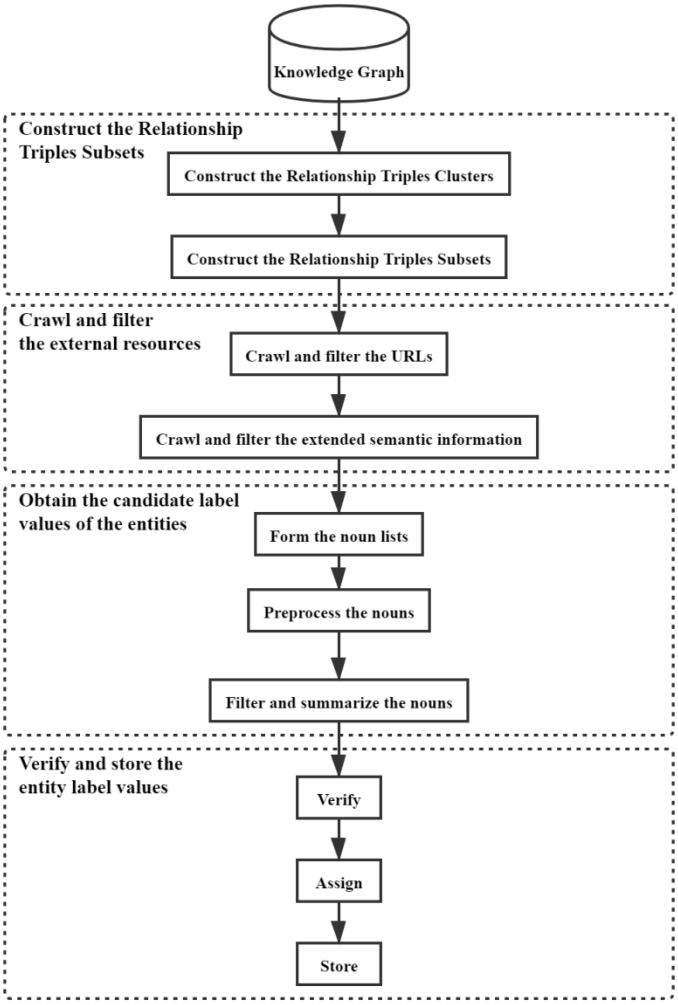

Figure 1. The overall framework of ELVAM

1) Construct the Relationship Triples Subsets. According to the relationship type in domain-specific Knowledge Graph, the triples with the same relationship are gathered to construct a Relationship Triples Cluster. A certain amount of triples is randomly selected from each cluster to construct a Relationship Triples Subset.

2) Crawl and filter the external resources. For the triples in each subset, the crawler framework Scrapy is used to crawl the URLs related to the entities of the triples in the external resource library, and F1-measure is utilized to filter the URLs in order to reduce noise data. After that, the extended semantic information of the entities is extracted from the external resource library according to the URLs, and the information is tailored to obtain the appropriate text.

3) Obtain the candidate label values of the entities. Firstly, we use natural language processing technology to extract all nouns that appear in the text to form a noun list. Secondly, we preprocess the nouns in the list by lemmatizing, merging the synonyms and abstracting the current words. Finally, we calculate the correlation ratio between Information Entropy and Conditional Entropy to summarize the nouns to obtain a candidate entity label value that is related to the Entity Role.

4) Verify and store the entity label values. After obtaining the candidate entity label values of each Relationship Triples Cluster, a certain amount of triples are randomly selected again to verify the label values. If the verification is successful, it means that the final entity label values are obtained, and the Label Triples Pattern of each Relationship Triples Cluster can be obtained by induction. The entities in each cluster are assigned label values according to the corresponding pattern, and complete entities data and relationship data are imported into the database.

3.2 Construct the Relationship Triples Subset

The goal of ELVAM is to obtain the label values, which is the ontology description in the domain, of each entity in Knowledge Graph. Each entity in domain-specific Knowledge Graph can be abstracted into an ontology. For example, the entity "Kismet" can be abstracted into the ontology "Film". In the triples with the same relationship type, all subjects have similar entity type distribution, which can be understood that the ontologies of all subjects have similar semantics. The same is true for the object (Miao, Fang, Song, Zheng, Fang, Meng, & Sun, 2016; Wu, Zhang, Deng, & Huai, 2019). Therefore, ELVAM does not need to perform label analysis on the subject or the object of each triples in Knowledge Graph. With the same relationship, the label value of a small amount of data can represent the label value of a large amount of data. Before constructing the subsets, it is necessary to construct the Relationship Triples Clusters. ELVAM can analyze the relationship types in Knowledge Graph and gather the triples of the same relationship to construct a Relationship Triples Cluster. After constructing, for each cluster, N triples are randomly selected to a Relationship Triples Subset.

3.3 Crawl and Filter the External Resources

In the Relationship Triples Subset, there are only triples with two entities and a relationship. When there is only an entity vocabulary but no entity extension semantic text, label analysis cannot be performed. Therefore, it is necessary to introduce the external resources. The external resources selected in this paper are Wikipedia.

In order to obtain the appropriate extended semantic information of the entity, we propose some searching patterns suitable for different Entity Roles. For the purpose of reducing the ambiguity, the entity is given some constraints, which are reflected on the searching patterns. As for a subject, there are three kinds of searching patterns: "subject", "subject+relationship" and "subject+relationship+object"; as for an object, there are also three kinds of searching patterns: "object", "relationship+object" and "subject+relationship+object". For example, assuming the triples "(Anna Karenina, directed_by, Bernard Rose)", the searching patterns of the subject "Anna Karenina" are "Anna Karenina", "Anna Karenina+directed_by" and "Anna Karenina+directed_by+" Bernard Rose" respectively; and the searching patterns of the object "Bernard Rose", are "Bernard Rose", "directed_by+Bernard Rose" and "Anna Karenina+directed_by+Bernard Rose" respectively.

After utilizing the searching patterns to explore in the external resource library, the external resources crawled and filtered in this article have two forms: URL form and TEXT form. Among them, URL is the identifier of the entity (Martinez-Rodriguez, López-Arévalo, & Rios-Alvarado, 2018), and TEXT provides the extended semantic information related to the entity. The relationship between URL and TEXT is that URL is the entrance to TEXT. We combine the crawler framework Scrapy and the proposed searching patterns to crawl URLs. In the end, an entity can get the URL results returned from the three searching patterns. These URL results are on the first page of the search results page.

For the same Relationship Triples Subset, after completing the crawling of the URL external resources, the Initial URL Library is formed. The Initial URL Library is filled with the information "(entity, URL, frequency)", where

entity refers to a subject or a object in the same subset and frequency refers to the number of occurrences (less than or equal to 3) of the same URL obtained from the three searching patterns related to the entity. In the Initial URL Library, an entity usually corresponds to multiple URLs, which are huge in number but inferior in quality. So filtering the URLs in the initial library is required. The workflow of filtering the URLs is shown in Figure 2.

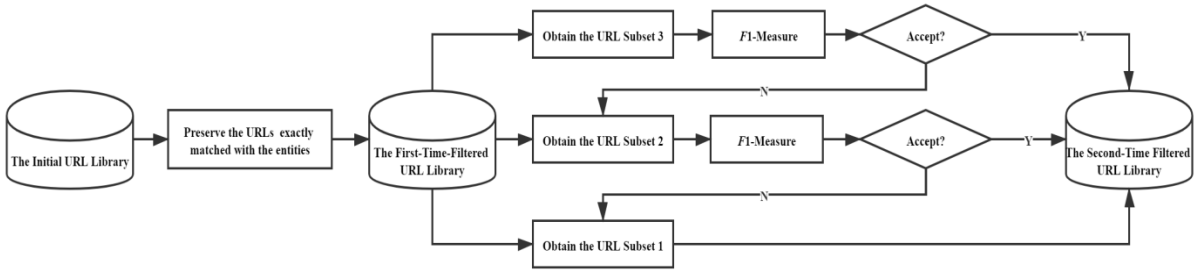

Figure 2. The workflow of filtering the URLs

There is something should be noted. N-URL refers to the URL with a frequency of n in the URL library, where n=1,2,3. Let us take the corresponding URLs to the entity "Anna Karenina" as examples to illustrate this. The URL "/wiki/Anna_Karenina_(1997_film)" [1] has a frequency of 3, so it is called 3-URL. The URL "/wiki/Anna_Karenina_(musical)" has a frequency of 2, so it is called 2-URL. The set formed by n-URL elements is called n-URL Set. The URL Subset 3 includes a 3-URL Set. The URL Subset 2 includes a 3-URL Set and a 2-URL Set. The URL Subset 1 is the URL sets in the First-Time-Filtered URL Library.

In Figure 2, the Initial URL Library needs to be filtered twice to get the Second-Time-Filtered URL Library. In the first filtering, the URLs that exactly match the entities are reserved to build the First-Time-Filtered URL Library. Let us continue to take the corresponding URLs to the entity "Anna Karenina" as examples. On the one hand, the URL "/wiki/Anna_Karenina" will be reserved, which is exactly matched the entity "Anna Karenina". On the other hand, the URL "/wiki/Tom_George_Kolath" will be discarded because of its totally mismatch with the entity "Anna Karenina". In addition, the URL "/wiki/Adaptations_of_Anna_Karenina" partially matched the entity "Anna Karenina" is also filtered. However, the URLs containing "()" or "#" will be retained, such as the URL "/wiki/Anna_Karenina_ (1997_film)". Because the phrase in "()" or after "#" can be served as an attributive and play an important role on supplementary.

In the second filtering, 3-URLs are selected from the First-Time-Filtered URL Library to form the URL Subset 3. However, considering that there may be too few correct URLs and too many noisy URLs in the URL Subset 3, we plan to use F1-measure to determine whether to accept the currently obtained URL Subset 3 to the Second-Time-Filtered URL Library. Specifically, when F1-measure is greater than or equal to $\alpha(\alpha \in (0,1])$, we accept the URL Subset 3. Otherwise, 2-URLs are selected from the First-Time-Filtered URL Library to form the URL Subset 2 with 3-URLs. For the URL Subset 2, the involved process whether to accept it to the Second-Time-Filtered URL Library is the same with for the URL Subset 3. When the URL Subset 2 is not accepted, the URL Subset 1 is directly accepted into the Second-Time-Filtered URL Library.

F1-measure combines precision *P* and recall *R*. The calculation formula is shown in (1):

$$F1 = \frac{2PR}{P+R} \tag{1}$$

where precision *P* is the ratio of the correctly discovered answers over all the returned answers, and recall *R* is the ratio of the correctly discovered answers over all the golden standard answers.

Here, the correctly discovered answers refer to the URLs that are exactly matched with the entities in the URL Subset 3 or the URL Subset 2. But for an entity, no matter how many such URLs are, the number of the correctly discovered answer is still 1. The returned answers refer to the URLs in the URL Subset 3 or the URL Subset 2.

The extended semantic text describing the corresponding entity can be obtained through the corresponding URL

[1] The prefix "https://en.wikipedia.org" has been omitted.

in the Second-Time-Filter URL Library. Generally, in the extended semantic text, the abstract is an overview to the entity. What is more, the definition of the entity can be obtained through the abstract, which usually appears in the first sentence involved the entity name. In the next step, the definition of the entity is utilized as a text source to obtain the entity label values.

3.4 Obtain the Candidate Label Values of the Entities

In Section 3.3, the set of definitions of all entities (including subjects and objects) in each Relationship Triples Subset is obtained through crawling and filtering the URL and the TEXT external resources. The definition can reflect the extended semantics of the entity, and the entity label value is a noun word. Based on it, the Part-Of-Speech Tagging is used in ELVAM to extract the word whose Part-Of-Speech is annotated as "NN", "NNS", "NNP", or "NNPS" from the definition, and the noun lists concerning the entities semantics are attained. Among them, "NN" represents a common noun while "NNP" represents a proper noun, and the extra letter "S" illustrates that a word is in a plural form.

From the set of definitions, we have formed the noun lists. To ensure the quality of the nouns in the list, it is necessary to preprocess the nouns, which is divided into three steps:

1) Lemmatize all nouns in the list. We convert plural nouns to singular nouns.

2) Merge the synonyms. We replace all synonymous words with a word of a certain form by the synonym dictionary WordNet.

3) Abstract the current words. The goal of this research is to obtain the label values that can generalize a class of entities but are not too abstract, so that the entity label values of the Relationship Triples Subset can represent the entity label values of the Relationship Triples Cluster. Therefore, we introduce Ontology Category Hierarchy Graph, and use Information Entropy as well as Conditional Entropy, to determine whether a current word ought to be abstracted into its hypernym or not.

Figure 3 is an example of Ontology Category Hierarchy Graph, which reflects the hierarchical structure among ontologies. The smaller the number of layer, the more abstract the ontology.

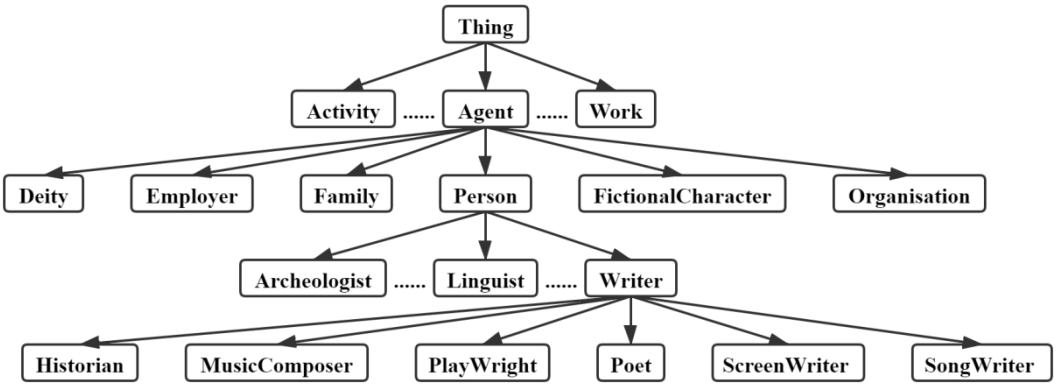

Figure 3. The example of Ontology Category Hierarchy

Information Entropy is defined as formula (2):

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i) \tag{2}$$

Where $P(x_i)$ represents the probability of random event X being x_i. While Conditional Entropy is defined as formula (3), which illustrates the expectation of random variable X to random variable Y under the condition of Y.

$$H(X \mid Y) = -\sum_{i}^{n} \sum_{j}^{n} P(x_i \mid y_j) \log_2 P(x_i \mid y_j) \tag{3}$$

Formula (4) is used to determine whether to abstract the current word to the upper word:

$$ratio(PW) = (H(PW) - H(PW \mid HW)) / H(HW) \quad (4)$$

Where PW represents the current word and HW represents its upper word. $H(PW)$ and $H(HW)$ refer to Information Entropy of the current word and the upper word respectively, indicating that how much the information contained. $H(PW \mid HW)$ refers to how much the information contained in the current word under the condition of the upper word. In formula (4), if $ratio(PW)$ is greater than the threshold $\lambda(\lambda \in (0,1))$, it means that the hypernym has provided explicit information for the current word. That is to say, the current word needs to be abstracted into its hypernym.

After the preprocessing steps of lemmatizing, merging the synonyms, abstracting the current words, the noun-frequency pairs sets are gotten. If you want to attain the candidate entity label values, you need to perform three rounds of screening on the noun-frequency pairs sets. The steps are as follows:

1) In the first round of screening, the maximum frequency of the noun in the noun-frequency pairs set should be obtained. For example, in the set $\{(n_1, f_1), (n_2, f_2),...,(n_k, f_k)\}$, if f_i is the maximum value among all f, then f_i is the maximum frequency. Afterwards, the nouns that appear more frequently than $\beta(\beta \in [\frac{1}{2}, 1))$ times the maximum frequency are kept and others are discarded. It is believed that the higher the frequency of a certain noun, the more likely it or its hypernym is the entity label value. Otherwise, the less likely it is.

2) In the second round of screening, the nouns with the same stem as the relationship type are retained as candidate entity label values, and the remaining nouns are discarded.

3) The third round of screening is performed when the candidate entity label values cannot be obtained in the second round of screening. With the help of the network corpus, we make statistics on the number of sentences that contain the Label Triples Pattern under the current noun. The more sentences and the higher the frequency of the noun, the more likely it is to be the correct Label Triples Pattern. Consequently, the noun can be determined as the candidate entity label value. For example, suppose there are two Label Triples Patterns (n_1, r, clv_o) and (n_2, r, clv_o), where n_1 and n_2 are the nouns being filtered now, r represents the relationship, and clv_o is the candidate label value for object. If there are more sentences under the condition of n_1 than under the condition of n_2, as well as the frequency of n_1 is higher than n_2, then n_1 is determined as the candidate entity label value.

At last, in order to obtain a label value that can generalize a class of entities but is not too abstract, it is necessary to determine whether the current candidate entity label value needs to be abstracted or not after three rounds of screening. If abstraction is required, the hypernym of the current word becomes the true candidate entity label value. If abstraction is not required, the current word is the true candidate entity label value. The specific algorithm named SAGA (Screening And Generalizing Algorithm) describing the above process is shown in Table 1.

Table 1. Screening And Generalizing Algorithm (SAGA)

Algorithm 1. Screening And Generalizing Algorithm (SAGA)

Input: a noun-frequency pairs set *NF*, a relationship in a Triples Subset *rel*

Output: a candidate label value *clv*

```
max = Obtain(NF)      // Get the maximum frequency
for each (n_i, f_i) in NF do
    if f_i >= (β * max) then
        if n_i <-samestem-> rel then      // n_i and rel have the same stem
            clv = n_i and goto 12
        end if
    else Delete((n_i, f_i))      // Delete (n_i, f_i) in NF
    end if
end for
Common(NF)      // Get the noun with the most sentences and the most frequent occurrences
clv = n_i, n_i is the most common one
DecideHypernym(clv)      // Decide whether to abstract clv to its hypernym or not
```

3.5 Verify and Store the Entity Label Values

The candidate entity label values can only become the final entity label values after successful verification. In terms of specific operations, the verification of ELVAM is as follows. For each Relationship Triples Cluster, *M* copies of data are randomly extracted to construct the Relationship Triples Subset. If the candidate result for *M* is consistent with the initial candidate result for *N*, it means that the verification is successful and the final entity label value is confirmed. Otherwise, the previous three modules of ELVAM are repeated until the final entity label value is determined.

After obtaining the final entity label values corresponding to the two Entity Roles, the Relationship Triples Cluster can be summarized into a Label Triples Pattern. According to the Label Triples Pattern, the label values are assigned to the corresponding entities in Knowledge Graph so as to obtain the entity-label value pairs.

Finally, we import the completed Knowledge Graph into Neo4j database (Chen, 2017) for other applications.

4. Experiment and Result Analysis

4.1 Experimental Data and Evaluation Criteria

The experiments are implemented on two real-world Knowledge Graph datasets: MetaKG from MetaQA (Zhang, Dai, Kozareva, Smola, & Song, 2018) and YagoKG from YAGO3-10 (Dettmers, Minervini, Stenetorp, & Riedel, 2018). The full name of MetaQA is "MoviE Text Audio QA", which is a question answering dataset that combines text data and audio data about movies. MetaKG, which supports MetaQA, is a movie knowledge base that originated from WikiMovies (Miller, Fisch, Dodge, Karimi, Bordes, & Weston, 2016). Its three types of triples do not carry out Entity Label Value Assignment Experiment, because the entities of these triples are very special and there is no rule. They have no meaning in assigning label values. YAGO3-10 is a subset of YAGO3, which comes from Wikipedia, WordNet and GeoNames (Mahdisoltani, Biega, & Suchanek, 2014). We construct a small dataset YagoKG related to the art field from YAGO3-10. The detailed statistics of the two datasets are shown in Table 2.

Table 2. The statistics of the datasets

Dataset	The number of triples	The number of entity	The number of relationship
MetaKG	105,347	38,343	6
YagoKG	15,890	7,145	6

In order to evaluate the performance of ELVAM model, we have gotten the sets of entity-label value pairs of MetaKG and YagoKG. The actual values in the set of entity-label value pairs for YagoKG are from ("YAGO3"), retaining only the representative nouns. Different from YagoKG, the actual values in the set for MetaKG come from MetaQA. In addition to providing the question-answering pairs, MetaQA also provides the question types. Based on the question types, the truth type (i.e., label value) of the answer (i.e., entity) to the question can be known.

Considering that ELVAM and Tipalo have similar research backgrounds. They both use the natural language definitions to assign the entity label values. Therefore, we take Tipalo as the benchmark to compare the experimental results between ELVAM and Tipalo.

In the article, precision, recall and F1-measure are used as the evaluation criteria to prove the advantages of ELVAM compared with Tipalo. In addition, Macro-F1 and Micro-F1 are utilized as the other evaluation criteria to evaluate the effectiveness of ELVAM. Macro-F1 first calculates F1-measure of each category, and then calculates the arithmetic average of F1-measure of all categories. While Micro-F1 is to calculate the overall precision and the overall recall first, and to calculate the overall F1-measure next, without distinguishing the categories.

4.2 Experimental Results

The parameters in ELVAM are set as follows.

- For each Relationship Triples Cluster in MetaKG, the number of triples randomly selected is $N = M = 10$.
- For each Relationship Triples Cluster in YagoKG, the number of triples randomly selected is $N = M = 20$.
- $\alpha = 0.6$. When F1-measure is greater than or equal to 0.6, the current URL Subset is accepted into the Second-Time-Filtered URL Library.

• λ = 0.3. When *ratio*(*PW*) is greater than 0.3, the current word should be abstracted to the upper word.

• β = 0.6. When the frequency of the noun in the list is less than 0.6 of the maximum frequency, there is no process will be done on it.

4.2.1 Comparison and Analysis of Results between ELVAM and Tipalo

In Tipalo, the types automatically assigned to the DBpedia entities are based on the natural language definitions from Wikipedia. The authors have randomly selected 627 Wikipedia pages for experiment, and have constructed a Golden Standard containing 100 entities for evaluation accordingly.

In order to reduce the error of the comparison between ELVAM and Tipalo, we use Tipalo model to conduct experiments on MetaKG and YagoKG.

Table 3. The comparison between ELVAM and Tipalo

Model	Precision	Recall	F1-measure
Tipalo (TS)	0.93	0.90	0.92
Tipalo (TS+WSD)	0.76	0.74	0.75
Tipalo (TS+WSD+TM_S)	0.62	0.60	0.61
Tipalo (TS+WSD+TM_D)	0.68	0.66	0.67
Tipalo_MetaKG	0.61	0.57	0.59
Tipalo_YagoKG	0.82	0.80	0.81
ELVAM_MetaKG	0.99	0.99	0.99
ELVAM_YagoKG	0.90	0.94	0.92

Table 3 illustrates the performance evaluation between ELVAM and Tipalo, where the results of Tipalo, including "Tipalo (TS)", "Tipalo (TS+WSD)", "Tipalo (TS+WSD+TM_S)", and "Tipalo (TS+WSD+TM_D)", are sourced from (Nuzzolese et al., 2013). The results based on MetaKG and YagoKG correspond to Tipalo_MetaKG and Tipalo_YagoKG, respectively. In Table 3, "Tipalo (TS)" means that the Tipalo model is composed of Type Selector (TS). "Tipalo (TS+WSD)" refers to that the Tipalo model is composed of Type Selector and Word Sense Disambiguation (WSD). "Tipalo (TS+WSD+TM)" means that the Tipalo model is composed of three parts: Type Selector, Word Sense Disambiguation and Type Matching (TM). The "S" and the "D" accompanied by "_" represents different mapping data resources.

According to the evaluation results in Table 3, it is found that F1-measure of ELVAM on MetaKG and YagoKG can reach more than 0.92, while F1-measure of Tipalo is lower than 0.92. Therefore, ELVAM is better than Tipalo on the aspect of assigning label values to entities.

It is observed that ELVAM crawls and filters the entity definitions in the external resource library. It extracts the nouns in the definitions and screens out them to obtain the entity label values. While Tipalo uses heuristics templates lack of flexibility to extract the definitions. It assigns the label values to the entities through RDFS inference, which can be wrong in some cases. ELVAM filters the extended semantic information of the entities by restricting searching patterns and calculating F1-measure, to reduce the ambiguity and improve the performance of label value assignment. While Tipalo does not consider the disambiguation of the entities when mapping to the Wikipedia pages, which may lead to poor performance on entity label value assignment.

4.2.2 Evaluation Results of ELVAM in Macro-F1 and Micro-F1

Table 4 is shown the evaluation results of ELVAM in the criteria of Macro-F1 and Micro-F1. The results show that ELVAM is effective for the problem of entity label value assignment.

Table 4. Evaluation results of ELVAM on different datasets

Dataset	Macro-F1	Micro-F1
MetaKG	0.8548	0.9997
YagoKG	0.7415	0.9444

In terms of Micro-F1, the results on the two datasets are above 0.90. In terms of Macro-F1, the result on MetaKG is 0.85 and the result on YagoKG is lower than 0.75. It may be that 299 entities lack the actual label values, which leads to the poor performance of ELVAM on YagoKG. Compared with Micro-F1, the results of Macro-F1 are relatively low, it is indicated that the number of triples contained in each Relationship Triples Cluster is unevenly distributed.

5. Discussion

5.1 The Limitations of Crawling

The limitations of crawling have the following aspects.

The first aspect is restriction. Because some websites are designed with anti-crawling mechanisms, it is difficult for us to get the contents in the websites. In this case, we may pay more attention to making a well crawler in order to crawl and attain the external resources. Otherwise, we are forced to collect the resources in the external library by manual.

The second aspect is pertinence. The crawler is not flexible enough. Once the structure of the web page changes, the parsing rules of the crawler may also be changed. That is to say, the crawler should be designed according to the web structure. When the external resource library is different, the crawler is different too.

5.2 The Advantages of ELVAM

When compared to Tipalo, ELVAM achieves the better evaluation results. In addition, when compared to other automatic assignment methods, ELVAM has possess of higher efficiency.

By predicting the label values of a small amount of the entities, we obtain the label values of all the entities in Knowledge Graph, which is the work done by ELVAM. Therefore, the granularity of the entity label values in Knowledge Graph is moderate after ELVAM processing. On the one hand, the predicted label values cannot be at a fine-grained level, because it is necessary to ensure that the small amount of label values are representative. On the other hand, if the predicted label values are at a coarse-grained level, it will affect the performance of the application of Knowledge Graph. Above all, the predicted entity label values should be with moderate granularity.

5.3 The Drawbacks of ELVAM

ELVAM is an entity label value assignment method based on domain-specific Knowledge Graph. It is also suitable for open-domain Knowledge Graph. Actually, open-domain Knowledge Graph is formed by a mixture of multiple specific domains. As long as you find the most distinguishable relationship that can divide the domains, starting from the subjects and the objects under these relationships, digging out the remaining relationships linked by the above subjects and the above objects, and gathering them together, domain-specific Knowledge Graph is constructed. In this way, we can use ELVAM to assign label values to each entity in the above Knowledge Graph. Therefore, Knowledge Graph for ELVAM can be extended from specific domain to open domain.

However, if the relationship that is the most distinguishable can not be found in the open domain, then it is a challenge to utilize ELVAM to entity label value assignment. In this case, the accuracy and the F1-measure may decrease, which results in ELVAM not being able to achieve the good performance in the open domain.

6. Conclusion

In this paper, Entity Label Value Assignment Method named ELVAM is proposed for the domain-specific Knowledge Graph, whose entity label values are nonexistent at the beginning. Firstly, according to the relationship types, the Relationship Triples Clusters are constructed and a Relationship Triple Subset of each cluster is collected. Secondly, for each subset, the extended semantic information of the subjects and the objects is obtained and gathered by using the external resource library. Thirdly, natural language processing technology is utilized to extract the nouns in the extended semantic information to form the noun lists. The nouns in the lists are preprocessed through the steps of lemmatizing, merging the synonyms, and abstracting the current words. In order to obtain the candidate entity label values with appropriate granularity related to Entity Role, three rounds of screening and a round of generalization is processed. Finally, after the successful verification, the Label Triples Pattern of each Relationship Triples Cluster is summarized. According to the patterns, the corresponding entities in Knowledge Graph are assigned label values, so as to achieve the purpose of assigning the label values to all entities in Knowledge Graph after performing entity label analysis on a relatively small amount of triples.

The method we propose can assign corresponding label values to entities in Knowledge Graph, which is an

extension. In future work, we intend to use the expanded Knowledge Graph in applications such as the question answering system, and utilize entity label values to improve the efficiency and the accuracy of the actual system.

Acknowledgments

This work is supported partly by Department of Education of Guangdong Province, and by Guangzhou Science and Technology Innovation Committee.

References

Chen, S. (2017). *Neo4j full stack development*. Beijing: Publishing House of Electronics Industry.

Dettmers, T., Minervini, P., Stenetorp, P., & Riedel, S. (2018, April). Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).

Entity. (2020, July 15). In Wikipedia, the free encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Entity

Knowledge Graph. (2020, July 15). In *Wikipedia, the free encyclopedia*. Retrieved from https://en.wikipedia.org/wiki/Knowledge_Graph

Liu, Q., Li, Y., Duan, H., Liu, Y., & Qin, Z. (2016). A survey of knowledge graph construction. *Journal of Computer Research and Development*, *53*(3), 582.

Mahdisoltani, F., Biega, J., & Suchanek, F. (2014). Yago3: A knowledge base from multilingual wikipedias. In *7th biennial conference on innovative data systems research*. CIDR Conference.

Martinez-Rodriguez, J. L., López-Arévalo, I., & Rios-Alvarado, A. B. (2018). Openie-based approach for knowledge graph construction from text. *Expert Systems with Applications*, *113*, 339-355. https://doi.org/10.1016/j.eswa.2018.07.017

Miao, Q., Fang, R., Song, S., Zheng, Z., Fang, L., Meng, Y., & Sun, J. (2016, October). Automatic identifying entity type in linked data. In *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation: Posters* (pp. 383-390).

Miller, A., Fisch, A., Dodge, J., Karimi, A. H., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*. https://doi.org/10.18653/v1/D16-1147

Miller, G. A. (1998). *WordNet: An electronic lexical database*. MIT press.

Moon, C., Jones, P., & Samatova, N. F. (2017, November). Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on conference on information and knowledge management* (pp. 2215-2218). https://doi.org/10.1145/3132847.3133095

Neelakantan, A., & Chang, M. W. (2015). Inferring missing entity type instances for knowledge base completion: New dataset and methods. *arXiv preprint arXiv:1504.06658*. https://doi.org/10.3115/v1/N15-1054

Nuzzolese, A. G., Gangemi, A., Presutti, V., Draicchio, F., Musetti, A., & Ciancarini, P. (2013, May). Tìpalo: A tool for automatic typing of dbpedia entities. In *Extended Semantic Web Conference* (pp. 253-257). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-41242-4_34

Ontology Classes. (2020, July 15). In *DBpedia*. Retrieved from http://mappings.dbpedia.org/server/ontology/classes/

Paulheim, H., & Bizer, C. (2013, October). Type inference on noisy rdf data. In *International semantic web conference* (pp. 510-525). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-41335-3_32

Scrapy Tutorial. (2020, July 15). In *Scrapy*. Retrieved from https://docs.scrapy.org/en/latest/intro/tutorial.html

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, *27*(3), 379-423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Wu, J., Zhang, R., Deng, T., & Huai, J. (2019, August). Named Entity Recognition for Open Domain Data Based on Distant Supervision. In *China Conference on Knowledge Graph and Semantic Computing* (pp. 185-197). Springer, Singapore. https://doi.org/10.1007/978-981-15-1956-7_17

YAGO3. (2020, July 15). In *The DBLP Computer Science Bibliography*. Retrieved from https://datahub.io/collections/yago

Zhang, Y., Dai, H., Kozareva, Z., Smola, A., & Song, L. (2018, April). Variational reasoning for question answering with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).

Zheng, W., Cheng, H., Yu, J. X., Zou, L., & Zhao, K. (2019). Interactive natural language question answering over knowledge graphs. *Information sciences*, *481*, 141-159. https://doi.org/10.1016/j.ins.2018.12.032