Big Data Research 22 (2020) 100159

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

Richpedia: A Large-Scale, Comprehensive Multi-Modal Knowledge Graph ☆

Meng Wang [a,b], Haofen Wang [c,*], Guilin Qi [a,b,**], Qiushuo Zheng [d]

[a] School of Computer Science and Engineering, Southeast University, Nanjing, China
[b] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China
[c] Intelligent Big Data Visualization Lab, Tongji University, Shanghai, China
[d] School of Cyber Science and Engineering, Southeast University, Nanjing, China

ARTICLE INFO

Article history:
Received 14 June 2020
Received in revised form 7 August 2020
Accepted 3 September 2020
Available online 15 October 2020

Keywords:
Knowledge graph
Multi-modal
Wikidata
Ontology

ABSTRACT

Large-scale knowledge graphs such as Wikidata and DBpedia have become a powerful asset for semantic search and question answering. However, most of the knowledge graph construction works focus on organizing and discovering textual knowledge in a structured representation, while paying little attention to the proliferation of visual resources on the Web. To consolidate this recent trend, in this paper, we present Richpedia, aiming to provide a comprehensive multi-modal knowledge graph by distributing sufficient and diverse images to textual entities in Wikidata. We also set Resource Description Framework links (visual semantic relations) between image entities based on the hyperlinks and descriptions in Wikipedia. The Richpedia resource is accessible on the Web via a faceted query endpoint, which provides a pathway for knowledge graph and computer vision tasks, such as link prediction and visual relation detection.

1. Introduction

With the rapid development of Semantic Web technologies, various knowledge graphs are published on the Web using Resource Description Framework (RDF), such as Wikidata [1] and DBpedia [2]. Knowledge graphs make RDF links among different entities available to construct a large heterogeneous graph, which facilitates to support semantic search [3], question answering [4] and other intelligent services. Meanwhile, public accessibility of visual resource collections has attracted much attention towards different Computer Vision [5] (CV) research purposes, including visual question answering [6], image classification [7], object and relationship detection [8], etc. We have witnessed promising results by encoding entity and relation information of textual knowledge graphs for CV tasks. Whereas, most knowledge graph construction work in the Semantic Web and Natural Language Processing (NLP) [9–11]communities still focus on organizing and discovering textual knowledge only in a structured representation. There remains a relatively large scope of utilizing visual resources for knowledge graph (KG) research. A visual database is commonly defined as a rich source of image or video data and feeds sufficient visual information about entities to KGs. Obviously, making link prediction and entity alignment in a wider scope can empower models to make better performance when considering textual and visual features simultaneously.

In order to highlight the advantages of Semantic Web to the academic and industry community, a number of KGs have been constructed over the last years, such as Wikidata [1] and DBpedia [2]. These datasets make the semantic relationships and exploration of different entities possible. However, there are too few visual sources within these textual KGs. Visual question answering and image classification performance are consolidated by several methods [6,12–14]. They have been developed for connecting textual facts and visual resources, but the underlying RDF links [15] from different entities and images to objects in the same image are still very limited. Hence, a scarcity of existing data resources is not capable of bridging the gap between visual resources and textual knowledge graphs.

As mentioned above, general knowledge graphs focus on the textual facts. At present, there is a lack of the complete multi-modal knowledge graph in academic community, which will hinder the future research of multi-modal fusion. To extend the capac-

☆ Permanent URL: http://richpedia.cn. Github Repository: https://github.com/wangmengsd/richpedia.

* Corresponding author at: Intelligent Big Data Visualization Lab, Tongji University, Shanghai, China.
** Corresponding author at: School of Computer Science and Engineering, Southeast University, Nanjing, China.
E-mail addresses: carter.whfcarter@gmail.com (H. Wang), gqi@seu.edu.cn (G. Qi).

https://doi.org/10.1016/j.bdr.2020.100159
2214-5796/

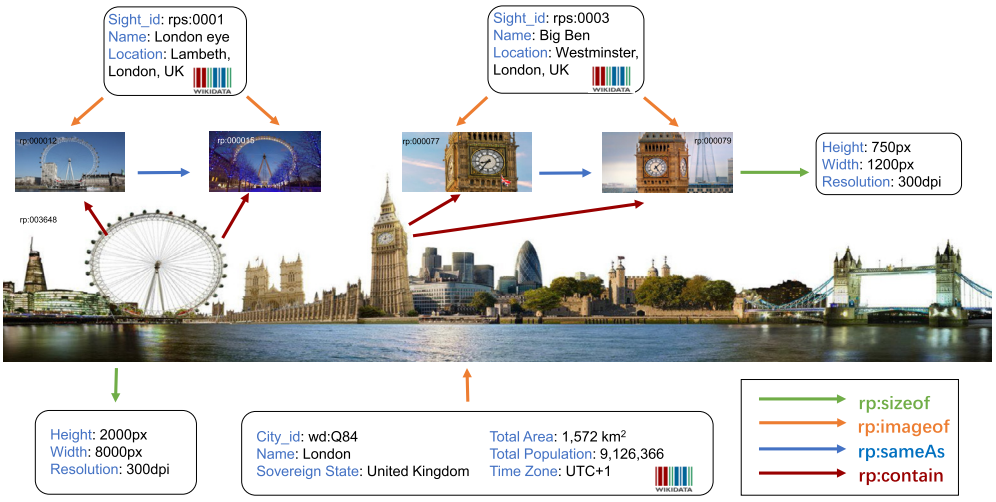

Fig. 1. Graphical illustration of multi-modal knowledge graph. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

ity of knowledge graphs, we provide a comprehensive multi-modal dataset (called Richpedia) in this paper, as shown in Fig. 1.

In summary, our Richpedia data resource generally makes the following contributions:

- To our best knowledge, we are the first to infuse comprehensive visual-relational resources into general knowledge graphs. Hence, we establish a big and high-quality multi-modal knowledge graph dataset, which offers a wider data scope to the researchers from The Semantic Web and Computer Vision.
- We propose a novel framework to construct the multi-modal knowledge graph. The process starts by collecting entities and images from Wikidata, Wikipedia, and Search Engine respectively. Images are then filtered by a distinctive retrieval model. Finally, RDF links are assigned between image entities based on the hyperlinks and descriptions in Wikipedia.
- We publish the Richpedia as an open resource, and provide a faceted query endpoint using Apache Jena Fuseki.[1] Researchers can retrieve and leverage data that distributed over general KGs and image resources to answer more richer visual queries and make multi-relational link predictions.

The rest of this paper is organized as follows. Section 2 describes the construction details of the proposed dataset. Section 3 describes the overview of the Richpedia ontology. The statistics and evaluation are reported in Section 4. Section 5 composes related work and finally, Section 6 concludes the paper and identifies topics for further work.

2. Richpedia construction

A knowledge graph (KG) can often be viewed as a large-scale multi-relational graph consisting of different entities and their relations. We follow the RDF model [15] and introduce the definition of the proposed multi-modal knowledge graph, Richpedia, as follows:

Richpedia definition: Let $\mathcal{E} = \mathcal{E}_{\mathcal{KG}} \cup \mathcal{E}_{\mathcal{IM}}$ be a set of general KG entities $\mathcal{E}_{\mathcal{KG}}$ and image entities $\mathcal{E}_{\mathcal{IM}}$, $\mathcal{R}$ be a set of relations between entities. $\mathcal{E}$ and $\mathcal{R}$ will be denoted by IRIs (Internationalized Resource Identifiers).[2] Let $\mathcal{L}$ be the set of literals (denoted by quoted strings, e.g. "London", "750px"), and $\mathcal{B}$ be the set of blank nodes. A Richpedia triple $t = \langle subject, predicate, object \rangle$ is a member of set $(\mathcal{E} \cup \mathcal{B}) \times \mathcal{R} \times (\mathcal{E} \cup \mathcal{L} \cup \mathcal{B})$. Overall, the multi-modal KG, Richpedia, is a finite set of Richpedia triples.

Fig. 2 illustrates the overview of Richpedia construction pipeline, which mainly includes three phases: **data collection** (described in Section 2.1), **image processing** (described in Section 2.2) and **relation discovery** (described in Section 2.3).

2.1. Data collection for Richpedia

Different from general KGs, we aim to construct a multi-modal dataset which constitutes rich image entities and their relations. Therefore, we consider following sources to populate Richpedia.

- Wikidata[3] is becoming an increasingly essential knowledge graph in the research community. We collect the KG entities from Wikidata as $\mathcal{E}_{\mathcal{KG}}$ in Richpedia.
- Wikipedia[4]: Wikipedia contains images for KG entities in Wikidata and a number of related hyperlinks among these entities. We will collect part of the image entities from Wikipedia and relations between collected KG entities and image entities. We will also discover underlying relations between image entities based on the hyperlinks and related descriptions in Wikipedia.
- Google,[5] Yahoo,[6] and Bing[7] image sources: We implemented a web crawler to gather sufficient image entities related to each KG entity. The web crawler will operate with the image search engine to capture input image entities which will be parsed query results. We employed Google Images, Bing Images, and Yahoo Image Search as image engines.

According to the Richpedia definition, we need to extract two types of entities (KG entities $\mathcal{E}_{\mathcal{KG}}$ and image entities $\mathcal{E}_{\mathcal{IM}}$), and generate Richpedia triples.

Entity IRI creation: Wikidata already contains unique IRI for each KG entity, we add these IRIs to Richpedia as the KG entities. In the current version, we mainly collected 30,638 entities about cities, sights, and celebrities. With these IRIs, the attribute information,

[1] https://jena.apache.org/documentation/fuseki2/index.html.
[2] https://www.w3.org/TR/rdf11-concepts/#dfn-iri.
[3] https://www.wikidata.org/wiki/Wikidata:Main_Page.
[4] https://www.wikipedia.org/.
[5] https://www.google.com/.
[6] https://search.yahoo.com/.
[7] https://www.bing.com/.

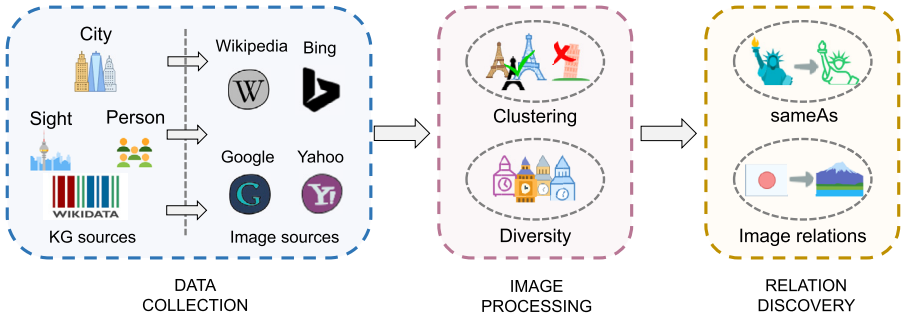

Fig. 2. Overview of Richpedia construction pipeline.

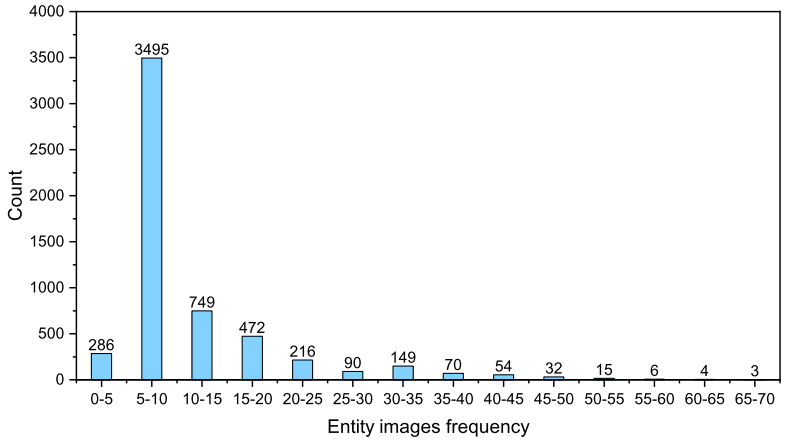

Fig. 3. The entity image frequencies in Wikipedia. There is a large portion of entities that only have a few images.

i.e., knowledge facts, of these KG entities can be directly queried on Wikidata.

For image entities, we can collect images directly from Wikipedia and create corresponding IRIs in Richpedia. However, as shown in Fig. 3, a large portion of images for KG entities are associated with long-tail. In other words, each KG entity will have very few visual information in Wikipedia. Therefore, as mentioned above, we obtained sufficient images from open sources and filtered out final image entities (details will be given in Section 2.2). After that, we will create IRIs for each image entity. In the current version, we have collected 2,883,162 images entities and kept 99.2 images per entity on average.

Triple generation: In Richpedia, we zoom in creating these three types of triples as follows:

$\langle e_i, rp{:}\text{imageof}, e_k\rangle$ indicates that an image entity e_i is an image of a KG entity e_k. An example is

$$\langle rp{:}001564, rp{:}\text{imageof}, wd{:}Q3130\rangle,$$

where *rp*:001564 is an image entity, i.e., a picture of *Sydney*, and *wd*:Q3130 is the KG entity in Wikidata.

$\langle e_i, rp{:}\text{attribute}, l\rangle$ indicates the visual feature (*rp*:attribute and numerical values l) of an image entity e_i. An example is

$$\langle rp{:}001564, rp{:}\text{size}, 700 * 1600\rangle.$$

where *rp*:001564 is a picture of *Sydney* attached with pixel information $700 * 1600$.

$\langle e_i, rp{:}\text{relation}, e_k\rangle$ establishes the semantic visual relations (*rp*:relation) between two image entities. An example is

$$\langle rp{:}000001, rp{:}\text{contain}, rp{:}000002\rangle.$$

where *rp*:000001 is a picture of *London* and it contains another image entity *London Eye* *rp*:000002 at the same time.

Since each IRI is unique, we can simply generate the triple link the $\langle e_i, rp{:}\text{imageof}, e_k\rangle$ and $\langle e_i, rp{:}\text{attribute}, l\rangle$ when processing data collection. For the triple $\langle e_i, rp{:}\text{relation}, e_k\rangle$, we will leverage related hyperlinks and text in Wikipedia to discover the relations (details will be discussed in Section 2.3).

2.2. Richpedia images processing

After collecting image entities, we need to process and frame high-quality images. Since our data comes from open sources, they are ranked by search engines with the high relevance score to the query. From the perspective of the multi-modal knowledge graph, the ideal image entities to a KG entity are not only highly relevant, but also ideally diverse. For instance, Fig. 4 shows the image diversity, the left and the middle image entities should be saved in Richpedia whereas the right should be filtered. It is intuitive to utilize clustering based methods to achieve the image diversity screening. In this paper, we employ a simple but pragmatic method below.

Diversity image retrieval: Given image entities crawled from search engines for a KG entity, we first apply the clustering algorithm, K-means on visual features to filter out the noise image entities. The similarity between two clustered images is measured based on the feature described in the following equation (1). The similarity between two image entities is defined as the cosine similarity of two feature vectors:

$$sim(e_i, e_j) = \cos\left(H(e_i), H(e_j)\right), \tag{1}$$

where e_i, e_j are image entities. A vector of visual feature in image e_i extracted by the VGG-16 deep neural network is presented as $H(e_i)$. We choose the value of K by sum of the squared errors

Fig. 4. Example of image entity diversity.

(SSE). The calculation of SSE is as follows:

$$SSE = \sum_{k=1}^{K} \sum_{p \in C_k} |p - m_k|^2 \tag{2}$$

When K is less than the number of true clusters, an increase in K will greatly increase the degree of aggregation of each cluster, leading a big decline in SSE. When K reaches the number of true clusters, the degree of the increased aggregation can be obtained by K. The return will decrease rapidly, so the decline of SSE will decrease sharply. SSE will then incline as K increases continuously. We will choose K according to the local minima of the SSE. In our experiment, we set our K at the number of 6.

For each image cluster, we will collect top-20 images in the following process. First, the image with the highest visual score is selected as the top ranked image. The second image is the one which has the largest distance to the first image. The third image is chosen as the image with the largest distance to both two previous images. During the diversity image detection retrieval, we also generate $\langle e_i, rp{:}\text{attribute}, l\rangle$ triples for the given image entity to provide its visual features in Richpedia.

After the acquisition of the images, we need to compute some different visual descriptors, which can describe the pixel-level features of the selected images (for instance, gray distribution and texture information for images). We then use these descriptors to calculate the similarity between the images, where the similarity can be calculated by integrating the distance between different descriptors. There are descriptors we compute are the following:

- **Gradation Histogram Descriptor:** Gradation histogram is a statistic of gradation distribution. Gradation histogram refers to the frequency of the gray size of all pixels in the image. Gradation histogram is a function of gray level, which represents the number of pixels with a certain gray level in the image, the goal of Gradation histogram reflects the frequency of a certain gray level in the image. We transform the image from color to grayscale and use the OpenCV Library to calculate the *Gradation Histogram Descriptor*. Finally, each image entity generates a description vector with 256 dimensions.
- **Color Layout Descriptor:** *Color Layout Descriptor* reflects the composition distribution of colors in the image, where the color histogram can represent the features of the probability of each color in the image. Color histograms are insensitive to geometric transformations such as rotation of the observation axis, translation and scaling with little amplitude, and are reluctant to change image quality (such as blurring). Therefore, we can measure the differences in the global color distribution of the two images by comparing the differences in color histograms. We calculate the three channels (R, G, B) of the color histograms respectively.
- **Color Moment Descriptor:** *Color Moment* is a simple and effective method to represent color features, which are first moment (mean), second moment (variance) and third moment (skewness). Since the color information is mainly distributed in the low order moment, the first-order moment, second-order moment and third-order moment are sufficient to express the color distribution of the image. To be more specific, the color moment has been proved to be effective to represent the color distribution in the image. We calculate three moments of the *Color Moment*.
- **GLCM Descriptor:** Gray-Level Co-occurrence Matrix (GLCM) is a common method to describe texture by studying the spatial correlation of gray. As the texture is formed by the grayscale distribution appearing repeatedly in the spatial position, there will be a certain grayscale relationship between the two pixels separated by a certain distance in the image space, that is, the spatial correlation characteristics of the grayscale in the image. We compute the *GLCM Descriptor* of the images.
- **Histogram of Oriented Gradient Descriptor:** Histogram of Oriented Gradient (HOG) is a feature descriptor that is employed in the field of computer vision and image processing for target detection. This technique is used to calculate the statistics of the direction information of local image gradients. We extract the edges of the grayscale image by computing its gradient (using Sobel and Laplacian kernels).

2.3. Richpedia relation discovery

In this section, we introduce the process of triple $\langle e_i, rp{:}\text{relation}, e_k\rangle$ generation. It is hard to detect these semantic relations based on pixel features of different images directly. These collected images from open sources are naturally linked to the input crawling seeds, i.e, KG entities, and image entities from Wikipedia and Wikidata. Therefore, we can exploit relevant hyperlinks and text in Wikipedia to discover the semantic relations (*rp*:relation) between image entities. Next we take *rp*:contain and *rp*:nearBy as examples to illustrate how to discover semantic relations among image entity, for example, *Place de la Concorde*, *Obelisk of Luxor*, and *Fountain of River Commerce and Navigation*.

As shown in Fig. 5, images of *Place de la Concorde*, *Obelisk of Luxor*, and *Fountain of River Commerce and Navigation* are extracted from the *Place de la Concorde* Wikipedia article in Chinese. From the semantic visual perspective, we could find that *Place de la Concorde* contains *Obelisk of Luxor* and *Fountain of River Commerce and Navigation*, and *Obelisk of Luxor* is near to *Fountain of River Commerce and Navigation*. To discover these relations, we collect textual descriptions over these images and propose three effective rules to extract final relations:

*Rule*1: If there is one hyperlink in the description, its Wikipedia entity corresponds to the image entity with a high probability. We detect the keywords from the description by Stanford CoreNLP. Then, relation will be discovered by a string mapping algorithm between keywords and predefined relation ontology. For instance, if we get the word 'left' in the textual descriptions between the two entities, we will get the 'nearBy' relation.

*Rule*2: If there are multiple hyperlinks in the description, we detect the core KG entity based on the syntactic parser and the

Fig. 5. Example of Richpedia relations discovery.

syntactic tree. Then, we take the core KG entity as the input and reduce this case to *Rule*1.

*Rule*3: If there is no hyperlink pointing to other articles in the description, we employ the Stanford CoreNLP to find the corresponding KG entities attached with Wikipedia articles and reduce this case to *Rule*1 and *Rule*2. Because *Rule*3 relies on the NER results which have low quality than annotated hyperlinks, its priority is lower than the first two rules.

3. Details in building

3.1. Collection of city KG entities

Firstly, we extract enough city KG entities from Wikidata as city KG entities in Richpedia. Since Wikidata is more authoritative in the open-source knowledge graph and many types of research are based on the Wikidata knowledge graph, Wikidata has become one of the spotlights of semantic web research.

We use SPARQL structured query language to extract the city KG entity from Wikidata, and select the entity whose attribute is "city", so that we can get the names of all cities and their corresponding Wikidata identifiers. Since the number of entities in the SPARQL query is limited and the selection of representative city entities is also straightforward for subsequent image collection and processing, we select large and medium-sized cities from the city KG entity list obtained as the first version of the Richpedia City KG entity list. Finally, we select 507 representative city entities. We store the city name and Wikidata identifier of these city KG entities in a specific JSON file for standby. At the same time, for each city KG entity, we will collect its information in the relevant text knowledge graph in Wikidata, such as the country of each city, total urban area, total urban population and time zone, etc., which are also stored in the corresponding JSON file.

3.2. Collection of sight KG entities

When we get the whole city KG entity list, we can collect the sight KG entities according to the city KG entity list, and regard it as the sight KG entity in Richpedia. We collect city scenic spot information from Ctrip website,[8] because the Ctrip website is a highly-qualified tourism website, which has a lot of unstructured information about city scenic spots, so we choose Ctrip website.

We acquire necessary knowledge of sight KG entity from the semi-structured text knowledge of scenic spots, and carry out structural processing to supplement the information in Richpedia as the sight KG entity at the same time. We use city KG entities as the search seeds to obtain the top 30 sight KG entities of each city KG entity. Meanwhile, we can crawl the location, opening time and brief introduction of the relevant sight KG entity from the Ctrip website and store them in the JSON file for future use.

3.3. Collection of celebrity KG entities

The collection of celebrity KG entities mainly comes from the Wikidata knowledge graph. We collect the celebrity KG entity list from Wikidata. First of all, we use SPARQL structured query language to select the entity whose attribute is "human" in Wikidata for filtering. It is unrealistic to return all the person name entities in terms of time complexity, so we limit the number of searches to 25000. We filter the initial list of celebrity entities, remove the unqualified celebrity KG entities, and finalize the list of celebrity KG entities of Richpedia. Meanwhile, it crawls some attribute information of these entities, such as "date of birth", "gender" and "allocation", then stores the information in the JSON file for backup.

3.4. Collection of image entities corresponding to KG entities

So far, we have collected the list of KG entities required by the first version of Richpedia and the textual knowledge graph information related to KG entities. Next, we need to collect the image entities corresponding to KG entities according to the list of KG entities.

We choose Google, Yahoo, Bing image search engines and Wikipedia as image entity collection tools. We decide using three image search engines instead of a single search engine because we can use different search engines complementarily to meet the integrity of the knowledge graph. Wikipedia is chosen because it contains images of KG entities in Wikidata, as well as numerous related hyperlinks and descriptive information between these entities. We use the list of KG entities as input into the crawler procedure, which proceeds the browser automation test framework Selenium to collect and store corresponding image entities. For each KG entity, we collect 100 images from three image search engines respectively and store them in the files of corresponding KG entities. At the same time, we generate a unique identifier of each entity. For the data collection in Wikipedia, we crawl the images attached with hyperlinks and description in the corresponding context. We then store the information in the JSON file. Iterating through the above steps, the necessary image entity resources for the construction of Richpedia are collected. Since our images are obtained from search engines, different search engines may return duplicate images; indeed, some KG entity semantics may be more remote. It will lead that some returned search images may not meet our requirements. We use an image clustering algorithm to complete the image denoising task.

Before proceeding the image filtering algorithm, we collected a total of 75,000 city image entities, 139,000 sight image entities, and 3,012,000 celebrity image entities.

[8] https://dst.ctrip.com/.

Fig. 6. The structure of the VGG-16 model.

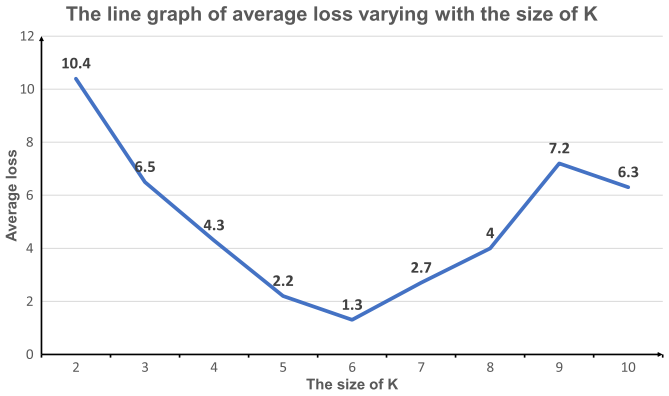

Fig. 7. The curve of the average clustering loss changing with the change of K values.

3.5. Filtering noise image entities

Some of the more remote entity entries that collected from web search engine may not return the image of the corresponding entity while searching, but the image of similar high-frequency entries may not be worthwhile. To guarantee the accuracy of our multi-modal knowledge graph, we need to make as many image entities as possible owned by all KG entities related to the corresponding KG entities. Hence, we designed an image clustering algorithm to filter these noise images.

We choose the unsupervised clustering algorithm to filter noise images, where the object of the specific clustering is the structural feature of each image entity. We use VGG-16 [16] to extract the structural features of the image, because there exists a disparity in the structure between the noise image entities and the related image entities. By clustering the structural features, we remove the noise image entities brought by the search engine. The structure of the VGG-16 we use is shown in Fig. 6. Since the feature vector generated in the last presentation layer of VGG-16 has a high dimension, we need to reduce the dimension of the generated feature vector to facilitate our subsequent clustering operations. We choose Principal Components Analysis (PCA) [17] to reduce the dimension, and finally represent the feature vector of the image entity as three-dimensional.

Among many image clustering algorithms, we choose the K-means algorithm to achieve our clustering filtering. First of all, we have too numerous image entities to make manual denoising appropriate, which is too time-consuming and labor-intensive. Moreover, the images we have collected so far do not have the label information, and selecting the recent popular deep learning network for labeling images will also be very exhausting. Building a gigantic neural networks with abundant layer regarding thousands of KG entities is not realistic for image denoising due to the wasteful computational cost. At the same time, we only have approximately 300 images in each training set. The training set is too small, and the training effect is not necessarily good. In conclusion, we choose the unsupervised learning K-means algorithm to realize the denoising process.

K-means algorithm is an unsupervised clustering algorithm. By the principle of similarity, the so-called clustering aims to divide data objects with higher similarity into the same cluster and data objects with higher dissimilarity into different clusters. This method will improve the similarity within the cluster as much as possible and reduce the similarity between clusters. One of the main performance factors of K-means is the selection of K value. We test the K value through experiments to select the most appropriate K value. We selected 100 KG entities for the experiment, and drew the curve of the average clustering loss changing over different K values, as shown in Fig. 7. From the graph, we can see that the average clustering loss drops to the local minima when K equals 6, which in turn, becomes the parameter for the following clustering process.

The advantage of setting K value in this way is that K value is not too small, so that the image denoising effect cannot be achieved due to the problem that the noise cannot be separated from the required image entity. Also, K value is not too large so that we need the image entity error filtering since the clustering effect that is too scattered. If the number of image entities in the cluster is less than five, we will filter them out. Fig. 8 shows the clustering effect of *Beijing*, *Washington*, *Tokyo*, *Paris*, *Berlin* and *London*. The green and purple points in the image represent the noise points.

3.6. Diversity detection

After the denoising process, we can get the image entities based on the high correlation between the corresponding KG entities. This is the reason why we need to detect the diversity of image entities. To achieve better query results, the image search engine will score the returned images, preferentially return the images with higher scores, because the more similar images may be different or the same search engine. However, these images have similar scores in search engines, so they will be returned to search results together. But as a multi-modal knowledge graph, the most ideal image entity is not only related, but also reasonable and diverse. To sum up, we need to detect the diversity of image entities, filter out too similar image entities, and ensure the image diversity of Richpedia.

With the cosine similarity of image entity feature vector, we can get the visual similarity between image entities, and then get the clustering results. Our specific method of image entity selection is as follows: first, we select the root node of the clustering tree, that is, the image entity with the highest clustering score. Then we select the image entity with the lowest similarity with the previous image entity to ensure diversity, to traverse all clustering trees, and select 20 image entities for each correct clustering cluster.

3.7. Relation discovery

Relationship discovery is a key step in the construction of multi-modal knowledge graph. It uses unstructured information

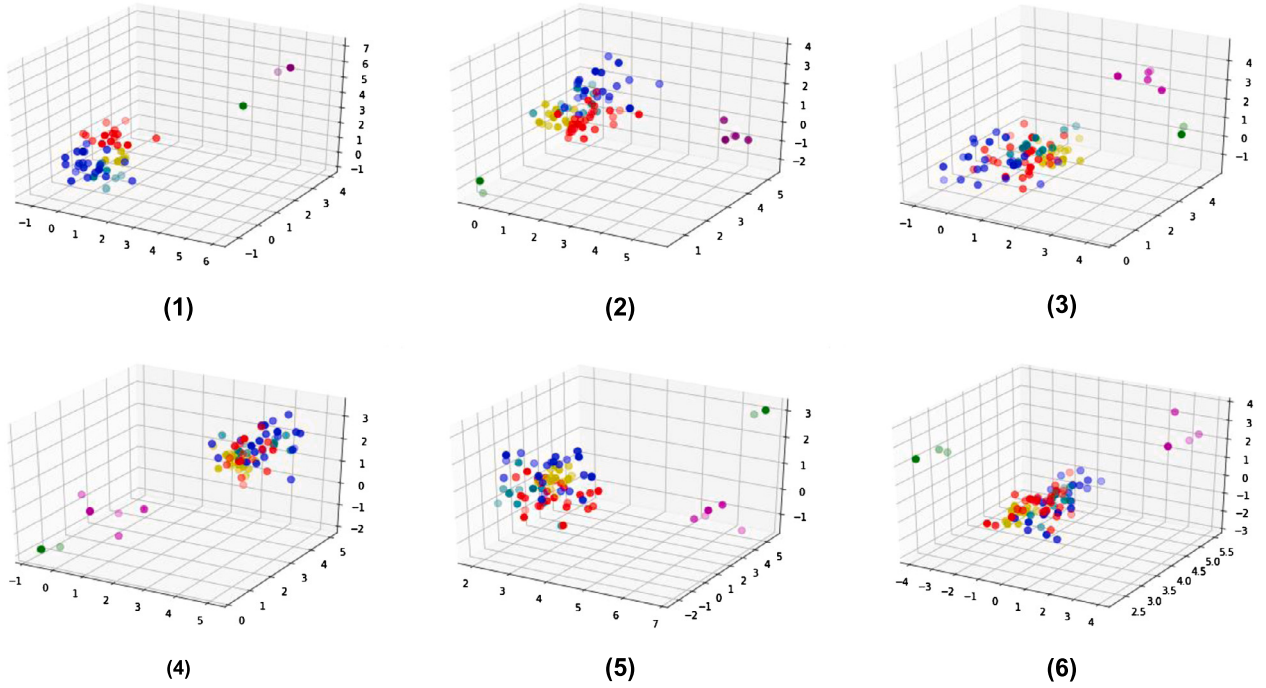

Fig. 8. The clustering effect of *Beijing*, *Washington*, *Tokyo*, *Paris*, *Berlin* and *London*.

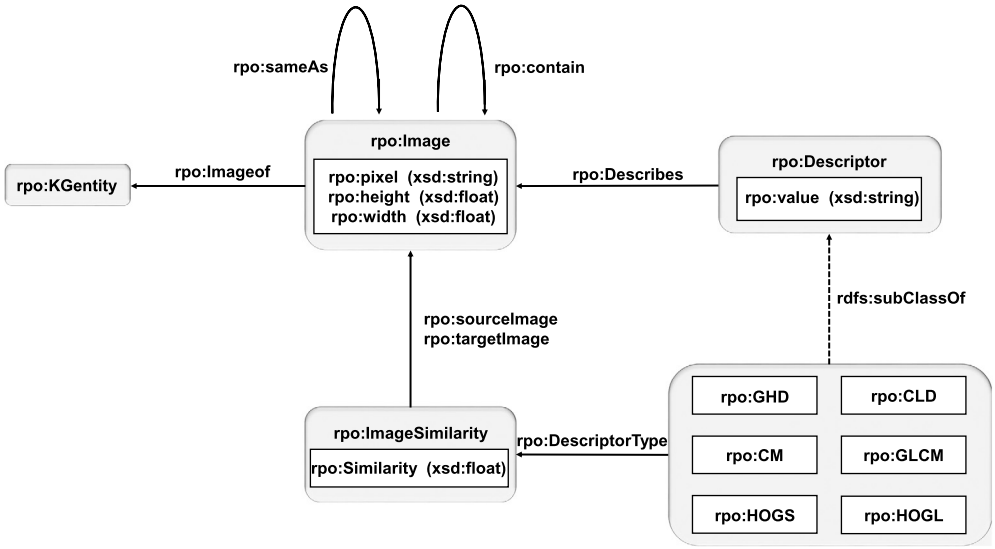

Fig. 9. Richpedia ontology overview.

to extract and infer the potential semantic relationship between image entities through NLP technology, establishes the connection relationship between scattered image entities, and realizes the relationship interconnection between entities.

The first category of relationship between the KG entity and the image entity is mainly established by the file structure in the Richpedia. Because the image entities are stored in the corresponding text knowledge graph entity articles, it is easy to generate such a relationship by using the file structure information.

The second category of relationship is the attribute value between image entity and image visual level information, which is mainly constructed by the visual features of any image entity, such as $rpo:height$, $rpo:width$ and so on.

The third kind of relationship is the visual semantic relationship between image entities. We mainly rely on image descriptions and hyperlink information to build the semantic relationship of image entities. The detailed rule description is shown in Section 2.3. We use NLP technology to extract the relationship and build a rich visual semantic relationship.

To sum up, we have established Richpedia through the above detailed steps, initially forming a relatively complete multi-modal knowledge graph, which provides a comprehensive dataset for later research.

4. Ontology

In this section, we describe the ontology we built for Richpedia, which consists of comprehensive image entities, multiple descriptors of image entities, and relationships among image entities. We create a custom lightweight Richpedia ontology to represent the data as RDF format. All the files formatted follows the N-Triples guidelines (https://www.w3.org/TR/n-triples/). All Rich-

```
@prefix    rpo:    <http://rich.wangmengsd.com/ontology/>
@prefix    rp:    <http://rich.wangmengsd.com/resource/>
rp:16.jpg    a    rpo:Image;
    rpo:Imageof    rps:0001;
    rpo:Height    600;    rpo:Width    900;
```

Fig. 10. RDF example of an image entity.

```
rp:16.jpg.GHD    a    rpo:GHD;
    rpo:Describes    rp:16.jpg;
    rpo:value    "[2823.0 ,  218.0 ,  256.0 ,  205.0 ,  ...]";
rp:16.jpg.CLD    a    rpo:CLD;
    rpo:Describes    rp:16.jpg;
    rpo:value    "[189.0 ,  62.0 ,  66.0 ,  49.0 ,  ...]";
rp:16.jpg.CM    a    rpo:CM;
    rpo:Describes    rp:16.jpg;
    rpo:value    "[64.8369 ,  92.1214 ,  195.548 ,  ...]";
rp:16.jpg.GLCM    a    rpo:GLCM;
    rpo:Describes    rp:16.jpg;
    rpo:value    "[0.0123 ,  0.0035 ,  0.0011 ,  ...]";
rp:16.jpg.HOGL    a    rpo:HOGL;
    rpo:Describes    rp:16.jpg;
    rpo:value    "[0.0666 ,  0.0120 ,  0.0033 ,  0.0012 ,  ...]";
```

Fig. 11. RDF example of the descriptors.

```
rp:gray_sim1    a    rpo:ImageSimilarity;
    rpo:sourceImage    rp:0.jpg;
    rpo:targetImage    rp:3997.jpg;
    rpo:Similarity    0.7750442028045654;
    rpo:DescriptorType    rpo:GHD;

rp:0.jpg    rpo:similar    rp:3997.jpg
```

Fig. 12. RDF example of a visual similarity relation.

pedia resources are identified under the http://rich.wangmengsd.com/resource/ namespace. The ontology is described at http://rich.wangmengsd.com/ontology/.

The overview of Richpedia ontology is shown by Fig. 9. The classes are displayed in the box. The solid edges represent the relation between instances of two classes, the dotted lines represent the relation between the classes themselves; for conciseness, the data type properties are listed in the class boxes.

A **rpo:KGentity** is an existing text knowledge graph entity that contains plenty of existing attribute information. A **rpo:Image** is an abstract resource representing an image entity of Richpedia dataset, describing the height and width of the image and the url of the image in the display website. The data types of **rpo:Height** and **rpo:Width** are both *xsd:float*. A **rpo:KGentity** links to many **rpo:Image**s by the relation of **rpo:imageof**. There exist some potential semantic relations between images, such as **rpo:sameAs**, **rpo:contain** and so on. In Fig. 10, we show the example of the RDF for the **rpo:Image** representation of *London Eye*.

A **rpo:Descriptor** is a visual descriptor of the image, linking to the **rpo:Image** by the relation **rpo:visual-describe**. A **rpo:Descriptor** has five subclasses, such as **rpo:GHD**, **rpo:CLD**, **rpo:CM**, **rpo:GLCM** and **rpo:HOG**. They describe the image in terms of grayscale, color, texture, edge and etc. For instance, in Fig. 11, there is a descriptor of the above image in terms of *London Eye*.

A **rpo:ImageSimilarity** is used to express the degree of similarity between images, it includes the **rpo:Similarity** which calculates between **rpo:sourceImage** and **rpo:targetImage** by the specified descriptor described before. **rpo:Similarity** represents the similarity of two images on the pixel level. We show the example of it in Fig. 12 in the next section.

Table 1
Statistics of entities.

	#KG entities	#Image entities
City	507	46,864
Sight	8,494	827,492
Person	20,984	2,040,414

Table 2
Statistics of Richpedia triples.

	#Overall
$\langle e_i, rp\text{:imageof}, e_k\rangle$	2,708,511
$\langle e_i, rp\text{:attribute}, l\rangle$	2,491,283
$\langle e_i, rp\text{:relation}, e_k\rangle$	114,469,776

5. Statistics and evaluation

In this section, we report the statistical data of Richpedia and a preliminary evaluation of its accuracy. At present, Richpedia includes 2.8 million entities and 172 million Richpedia triples.

The statistics of KG entities and image entities are depicted in Table 1. The numbers of the city entities, sight entities, and person entities in Richpedia are 507, 8,494, and 20,984. For image entities, there are 46,864 images of cities, 827,492 images of sights, and 2,040,414 images of famous people. The number of the Richpedia triples of three different types are depicted in Table 2. As shown in Table2, there are 2,708,511 relations between KG entities and image entities, 114,469,776 relations among image entities, and 2,491,283 relations between image entities and values such as pixel information.

To verify the effectiveness of the clustering algorithm and diversity detection, we manually construct 10,000 image distributions as ground-truths and compared it with the result of our diversity image retrieval model. The precision is 94% and the recall is 86%. The precision is the correct number of samples in the dataset after our algorithm divided by the total number of samples, and the recall is the correct number of samples in the dataset after our algorithm divided by the total number of samples in the ground-truth dataset. For the cause of images wrongly or missing distributed to entities, the reason is that we choose the image which has the largest distance to the collected image in the same cluster during the diversity image retrieval. Therefore, some noisy points may be misclassified in our final results to affect the precision then, and some images may not be ranked within the top-20 to affect the recall then.

As for the accessibility of data, due to the capacity limitation of our website server, we set up an online access platform that only displays some of the Richpedia data, but we provide Google Cloud Driver download link for all data. In the Google Driver download link, you can find the download link of the full data and the link of NT files referred to the data description, including the visual relations between the images, the image feature descriptors and so on. With respect to sustainability, because of the large size of the dump, we have not yet found a mirror host to replicate the data. Because we have a long-term plan for Richpedia, the dataset will be inactive maintenance and development. As for updating the dataset, although it is expensive to build the original dataset, we plan to implement incremental updates. The descriptors for these images can then be computed, while only the k-nn similarity relations involving new images (potentially pruning old relations) need to be computed.

Using the visual descriptors of image entities generated in Section 2.2, we design an experiment to calculate the similarity between image entities. First, we use the OpenCV library to calculate the visual descriptors for each image entity. Next, we use visual descriptors to calculate the similarity between images. For each

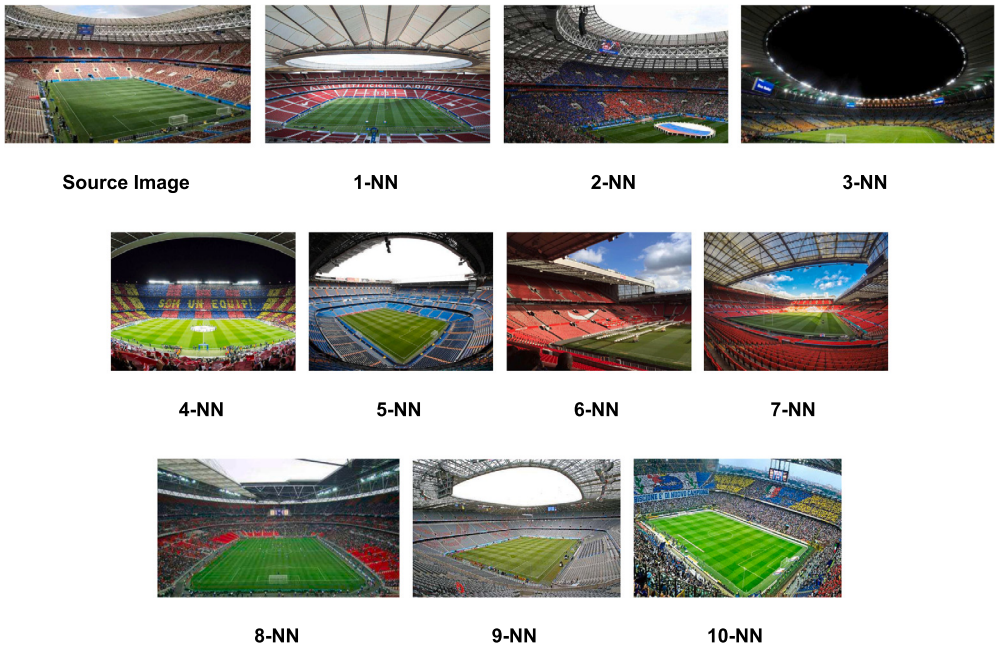

Fig. 13. 10 nearest neighbors of an image of *Russian Luzhniki Stadium* using HOG.

image entity, we calculate ten nearest neighbors for image entities according to each visual descriptor. For calculating the nearest neighbor image entities, we have the classical algorithm and boosted approximate NN matching algorithm.

The problem of nearest neighbor search is a major problem in many applications, such as image recognition, data compression, pattern recognition and classification, machine learning, document retrieval system, statistics and data analysis. However, solving this problem in high-dimensional space seems to be a very difficult task, and no algorithm is obviously superior to the standard brute force search. As a result, more and more people switch their attention towards a class of algorithms that perform the approximate nearest neighbor search. These methods have proved to be adequate approximation in many practical applications, which is much more efficient than the exact search algorithm. In computer vision and machine learning, finding the nearest neighbor in training data is expensive for a high-dimensional feature. For high-dimensional features, the randomized k-d forest[9] is the most effective method at present.

For the image entities of online access platform, we use the classical nearest neighbor algorithm to calculate the similarity between image entities due to the relatively small amount of image entities, tens of thousands approximately. The advantage of this algorithm is that it traverses all data sets, so it has relatively high accuracy and can perfectly reflect the similarity between image entities. However, the drawbacks of the classical nearest neighbor algorithm is explicit. For each image entity, we need to traverse all other image entities, this method belongs to brute force search, so it has high time complexity and will consume a lot of time and computing costs. But for the complete Richpedia dataset, if we want to calculate the similarity between image entities, we can only choose Fast Library for Approximated Nearest Neighbors (FLANN) since it has been proved to scale for large datasets. Although it will decrease inaccuracy, it is an optimal choice for large data sets in terms of integration accuracy and time complexity.

We design an experiment which contains 30,000 images. We configured FLANN with a goal precision of 95% and tested it on a brute-forced gold standard. First, we use the classical nearest neighbor algorithm to calculate, while it took 18 days to compute with 8 threads. When we test on the FLANN, it finished in 15 hours with 1 thread. Finally, FLANN achieved the precision of 76%. In Fig. 13, we show an example of similarity search results based on HOG descriptor, which captures information about edges in an image.

6. Use-cases

We first provide some examples of queries on the online access platform.

First, we can query the entity information in Richpedia, including image entities and KG entities. The first step is to select the entity category of the query, and then select the entity we want to query specifically. For example, in Fig. 14, if we want to query the KG entity information and image entity information of the city of *Ankara*, we can select the corresponding *Ankara* label in the dropdown selector. The upper half of the page that appears after that is the KG entity information of *Ankara*, and the lower half is the image entities of *Ankara*.

Second, we can query the visual semantic relation between image entities through Richpedia's online access platform. After we select the entity for querying, we can view the visual semantic relation of the image entity by clicking on the corresponding image entity. For example, when we want to query an image entity who has a **rpo:sameAs** relation with the *Beijing Zoo* image entity, we can click on the image entity and get the result as shown in the Fig. 15.

7. Related work

With the booming of artificial intelligence technology, knowledge graph, as the backbone of artificial intelligence, has escalated awareness from both academic and industrial areas owing to its powerful knowledge representation and reasoning capability. In recent years, knowledge graph has been widely used in semantic search, question and answer, knowledge management and other fields.

[9] A data structure that divides data points in k-dimensional space.

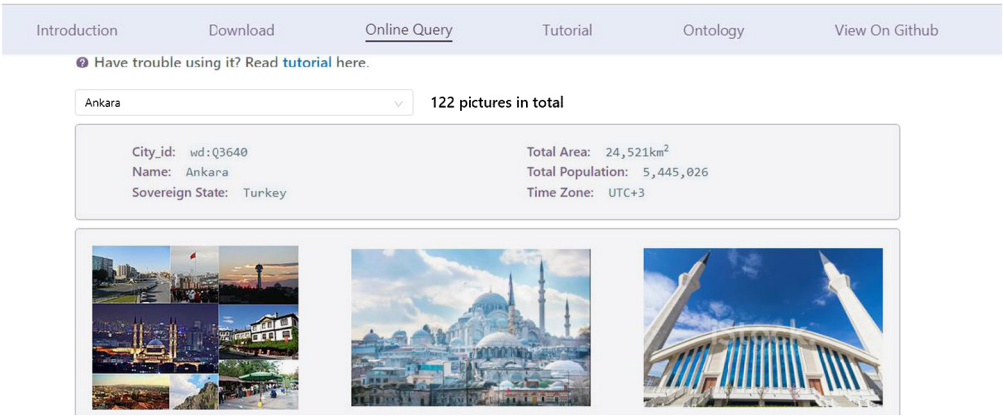

Fig. 14. Results of querying entity about *Ankara*.

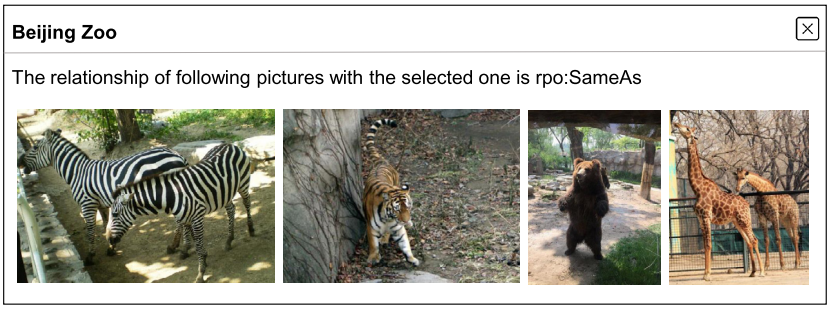

Fig. 15. Results of querying relations about Beijing Zoo entity.

Amongst the available datasets describing multimedia, the emphasis has been put on capturing the high-level metadata of the multimedia files (e.g., author, date created, file size, width, duration) rather than audio or visual features of the multimedia content itself [18,19]. Recently, several methods [6,12–14,20] have been developed for connecting textual facts and visual resources. IMGpedia [14] is a linked dataset that provides visual descriptors and similarity relationships for Wikimedia Commons. This dataset is also linked with DBpedia and DBpedia Commons to provide semantic context and further metadata.

Zhu et al. [6] exploited knowledge graphs for visual question answering for a specified purpose, generating a small amount of very distinctive images, and also proposed a knowledge base framework to handle all kinds of visual queries without training new classifiers for new tasks, these annotations represent the densest and largest dataset of image descriptions, objects, attributes, relationships, and question answers. Visual Genome dataset [12] aims at collecting dense image annotations of objects, attributes, and their relationships. Unfortunately, the visual relationship between IMGpeida [14] is limited to pixel level similarity, but not accessible for the relations between semantic, visual-relational facts in [12]. It is still incapable of providing relational information within the same image. For instance, a relation "nextTo" between *London Eye* and *River Thames* may be discovered as they appeared in the same image, but there is no further information about the relations between *London Eye* and *London*. MMKG [13] intended to perform relational reasoning across different entities and images in different knowledge graphs. However, it was constructed specifically for textual knowledge graph completion and focused on small datasets (FB15K, DBPEDIA15K, and YAGO15K). MMKG also did not consider the diversity of images when distributing images to relevant textual entities.

Nowadays, the knowledge graph has been prevalent in the fields of artificial intelligence, database, and semantic Web. It refers to any knowledge collection represented as a graph, such as a semantic Web knowledge base (e.g., DBpedia), an RDF dataset, and a formal ontology.

7.1. DBpedia

DBpedia [20] project is a community project, which aims to extract structured information from Wikipedia and make it accessible on the web. The generated DBpedia knowledge base currently describes more than 2.6 million entities. For each entity, DBpedia defines a unique global identifier, which can be dereferenced into a rich RDF descriptive entity on the network. DBpedia includes 30 human-readable language versions, forming a relationship with other resources. In the past few years, more and more data publishers began to set up data set links to DBpedia resources, making DBpedia a new data web interconnection center. At present, the Internet data source network based on DBpedia provides about 4.7 billion pieces of information, covering geographic information, people, companies, films, music, genes, drugs, books, science publishing houses, and other fields.

7.2. Wikidata

Wikidata is a knowledge graph jointly planned by Wikimedia Foundation (WMF) and the core project of the Wikimedia data management strategy. To make full use of Wikidata's potential, one of the main challenges is to provide reliable and powerful services for data sharing queries, for which WMF chooses to use semantic technology. Active SPARQL endpoints, regular RDF dumps, and linked data APIs are in many of the backbones uses that are now forming Wikidata.

Wikidata aims to overcome this inconsistency by creating new ways for Wikipedia to manage its data globally. The main achievements of Wikidata are as follows: Wikidata provides a free collaborative knowledge base that can be shared by all people; Wikidata soon becomes one of the most active projects of Wikimedia; more

and more other websites get content from Wikidata in every page view to increase the visibility and usefulness of big data.

7.3. IMGpedia

IMGpedia is a large-scale linked data set. It collects a large number of visual information from images in the Wikimedia Commons dataset. It constructs and generates 15 million visual content descriptors. There are 450 million visual similarity relationships between images. Besides, there are links between DBpedia and individual images.

IMGpedia aims to extract the relevant visual information from the images distributed by Wikipedia. We can collect the images of all terms and all multi-modal data (including author, date, size, etc.) from Wikimedia, and generate the corresponding image descriptors for each image. Linked data rarely considers multimedia information, but multimedia data is also an important part of the Internet. In order to explore the combination of link data and multimedia data, the author constructs IMGpedia, calculates the descriptors of images used in Wikipedia entries, and then links these images and their descriptions with the encyclopedia knowledge graph.

IMGpedia is a precedent of a multi-modal knowledge graph. It can be used to combine semantic knowledge graph and multi-modal data to confront challenges and opportunities. IMGpedia uses four image descriptors for benchmark testing, and the references and implementation of these descriptors are public. IMGpedia provides links to Wikidata. Since the categories in DBpedia are not appropriate for some visual semantic queries, IMGpedia aims to provide a better semantic query platform.

IMGpedia is an excellent precedent in multi-modal direction, but there are some problems, such as sparse relationship types, fewer relationship numbers, unclear image classification, etc., which are also the problems that try to be solved in this paper.

8. Conclusion and future work

This paper presents the process of constructing a multi-modal knowledge graph (Richpedia). Our work is to collect images from the Internet for entities in textual knowledge graphs. Then, images are filtered by a diversity retrieval model and RDF links are set between image entities based on the hyperlinks and descriptions in Wikipedia. The result is a big and high-quality multi-modal knowledge graph dataset, which provides a wider data scope to the researchers from The Semantic Web and Computer Vision. We publish the Richpedia as an open resource and provide a facet query endpoint. As future work, our plan is to broaden the type and scope of entities, so that Richpedia becomes more comprehensive and covers more topics.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests.

Acknowledgements

This work was supported by National Natural Science Foundation of China with Grant Nos. 61906037 and U1736204; National Key Research and Development Program of China with Grant Nos. 2018YFC0830201 and 2017YFB1002801; the Judicial Big Data Research Centre, School of Law at Southeast University with Grant No. 4313059291; the Fundamental Research Funds for the Central Universities with Grant No. 4009009106 and 22120200184.

References

[1] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Commun. ACM 57 (10) (2014) 78–85.
[2] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, Dbpedia-a crystallization point for the web of data, Web Semant. 7 (3) (2009) 154–165.
[3] W.-t. Yih, M.-W. Chang, X. He, J. Gao, Semantic parsing via staged query graph generation: question answering with knowledge base, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol. 1, 2015, pp. 1321–1331.
[4] P. Trivedi, G. Maheshwari, M. Dubey, J. Lehmann, Lc-quad: a corpus for complex question answering over knowledge graphs, in: International Semantic Web Conference, Springer, 2017, pp. 210–218.
[5] R.K. Heaton, PAR Staff, Wisconsin Card Sorting Test: Computer Version 2, Psychological Assessment Resources, Odessa, 1993, pp. 1–4.
[6] Y. Zhu, C. Zhang, C. Ré, L. Fei-Fei, Building a large-scale multimodal knowledge base system for answering visual queries, preprint, arXiv:1507.05670, 2015.
[7] K. Marino, R. Salakhutdinov, A. Gupta, The more you know: using knowledge graphs for image classification, in: CVPR, 2017, pp. 2673–2681.
[8] X. Liang, L. Lee, E.P. Xing, Deep variation-structured reinforcement learning for visual relationship and attribute detection, in: CVPR, 2017, pp. 848–857.
[9] C.D. Manning, C.D. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, 1999.
[10] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, D. McClosky, The Stanford corenlp natural language processing toolkit, in: ACL: System Demonstrations, 2014, pp. 55–60.
[11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (Aug) (2011) 2493–2537.
[12] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D.A. Shamma, et al., Visual genome: connecting language and vision using crowdsourced dense image annotations, Int. J. Comput. Vis. 123 (1) (2017) 32–73.
[13] Y. Liu, H. Li, A. Garcia-Duran, M. Niepert, D. Onoro-Rubio, D.S. Rosenblum, Mmkg: multi-modal knowledge graphs, preprint, arXiv:1903.05485, 2019.
[14] S. Ferrada, B. Bustos, A. Hogan, Imgpedia: a linked dataset with content-based analysis of wikimedia images, in: ISWC, Springer, 2017, pp. 84–93.
[15] W.W.W. Consortium, et al., Rdf 1.1 concepts and abstract syntax, 2014.
[16] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, preprint, arXiv:1409.1556, 2014.
[17] G.H. Dunteman, Principal Components Analysis, vol. 69, Sage, 1989.
[18] M. Addis, W. Allasia, W. Bailer, L. Boch, F. Gallo, R. Wright, 100 million hours of audiovisual content: digital preservation and access in the prestoprime project, in: INTL-DPIF, ACM, 2010, p. 3.
[19] K. Kurz, H. Kosch, Lifting media fragment uris to the next level, in: LIME/SemDev@ESWC, 2016, pp. 18–25.
[20] G. Vaidya, D. Kontokostas, M. Knuth, J. Lehmann, S. Hellmann, Dbpedia commons: structured multimedia metadata from the wikimedia commons, in: ISWC, Springer, 2015, pp. 281–289.