**Noname manuscript No.**
(will be inserted by the editor)

# Knowledge Graph Construction from Multiple Online Encyclopedias

**Tianxing Wu[1,2] · Haofen Wang[3] · Cheng Li[1] · Guilin Qi[1] · Xing Niu[4] · Meng Wang[1] · Lin Li[1] · Chaomin Shi[1]**

Received: date / Accepted: date

**Abstract** In recent years, lots of knowledge graphs built from Wikipedia, the largest multilingual online encyclopedia, have been published on the Web to support various applications. However, since non-English data in Wikipedia are sparse, some projects work on knowledge graph construction from multiple non-English online encyclopedias, but many technical details are missing, so it is hard to reuse their frameworks or techniques. In this paper, we propose a new framework to solve knowledge graph construction from multiple online encyclopedias. The core modules are knowledge extraction and knowledge linking. Knowledge extraction consists of regular extraction, i.e., extracting targeted article contents in the whole online encyclopedias periodically, and live extraction, which only extracts the article contents of new and updated entities. Knowledge linking utilizes heuristic lightweight entity matching strategies and a semi-supervised learning method to find duplicated entities and properties from different online encyclopedias. Experimental results show that our approaches for knowledge extraction and linking outperform state-of-the-art baselines in different evaluation metrics, and our framework can generate a large-scale knowledge graph after inputting multiple online encyclopedias.

**Keywords** Knowledge Graph · Knowledge Extraction · Knowledge Linking · Semantic Web

## 1 Introduction

With the development of Semantic Web, a growing amount of open structured (RDF) data has been published on the Web. Linked Data [2] initiates the ef-

✉ Corresponding Author: Guilin Qi (gqi@seu.edu.cn)
· [1]Southeast University, China
· [2]Nanyang Technological University, Singapore
· [3]Shanghai Leyan Technologies Co. LTD, China
· [4]University of Maryland, USA

fort to connect the distributed data across the Web and there have been over 1,200 datasets within Linking Open Data (LOD) community project[1]. The core datasets in LOD are the knowledge graphs built based on the multilingual online encyclopedia: Wikipedia, such as DBpedia [15], YAGO [17] and BabelNet [19]. These multi-domain encyclopedic knowledge graphs are important foundations of various intelligent applications, e.g., semantic search, question answering and domain-specific knowledge graph construction. However, non-English data in Wikipedia are sparse, which limits the development of non-English knowledge graphs. Actually, there exist many non-English online encyclopedias, such as Baidu Baike[2] (Chinese), Hudong Baike[3] (Chinese), Doosan Encyclopedia[4] (Korean), EcuRed[5] (Spanish) and etc. Therefore, to obtain more non-English knowledge, we study knowledge graph construction from multiple online encyclopedias in this paper.

Currently, only a few projects work on constructing knowledge graphs from multiple online encyclopedias, including Zhishi.me [24], CN-DBpedia [37] and XLORE [32]. One main problem existing in these publications is that many technical details are missing, so it is hard to reuse the proposed construction frameworks or techniques. For example, CN-DBpedia and XLORE do not introduce how to integrate knowledge from different Chinese online encyclopedias, i.e., Baidu Baike, Hudong Baike and Chinese Wikipedia[6]. XLORE and Zhishi.me do not provide the mechanisms of knowledge update, and although CN-DBpedia has an active update strategy, the details on how to implement this strategy on each online encyclopedia are still unclear.

Hence, we aim to present a framework, which clearly solves knowledge graph construction from multiple online encyclopedias. The core modules are knowledge extraction and knowledge linking. For knowledge extraction, we apply two strategies, i.e., regular extraction and live extraction. Regular extraction means extracting targeted article contents of all entities in online encyclopedias periodically. However, since a large proportion of knowledge is invariant (e.g., the birth place and birth date of a person), this strategy seems wasteful and may incur massive network overhead, which limits the update frequency and leads to knowledge obsolescence. To overcome the weakness and keep the freshness of the constructed knowledge graph, live extraction allows the knowledge graph to be up-to-date with a short latency. In live extraction, we mainly leverage the Wikipedia OAI-PMH live feed[7] which provides the update stream of Wikipedia, and a supervised update frequency predictor based on a Long Short-Term Memory (LSTM) Network [11], to help find and extract the entities with updated article contents as well as emerging entities.

[1] `http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData`
[2] `https://baike.baidu.com/`
[3] `http://www.baike.com/`
[4] `http://www.doopedia.co.kr/`
[5] `https://www.ecured.cu/`
[6] `https://zh.wikipedia.org/`
[7] `https://stream.wikimedia.org/v2/stream/recentchange/`

For knowledge linking, we first use three simple yet effective strategies to perform lightweight entity matching. Based on the resulting high-quality matched entities, we then propose a semi-supervised learning method to discover property-based rules to find equivalent entities from different online encyclopedias. With this proposed entity matching method, we can get not only large-scale matched entities, but also equivalent properties for free.

After applying our proposed framework of knowledge graph construction to three largest Chinese online encyclopedias, i.e., Baidu Baike, Hudong Baike and Chinese Wikipedia, we acquire a new Chinese knowledge graph, called Zhishi.me2, which is a new version of the Zhishi.me project. Compared with Chinese DBpedia and Chinese YAGO, Zhishi.me2 has much more Chinese entities and Chinese entity facts, which shows that mining multiple online encyclopedias is important and valuable for non-English knowledge graph construction.

In summary, the main contributions of this work are listed as follows:

– We propose a new general framework to knowledge graph construction from multiple online encyclopedias to facilitate non-English knowledge acquisition. Technical details of each part in the framework are clearly provided.
– In our framework, we present a new live extraction method to keep knowledge from being outdated, and a general semi-supervised rule learning method to find not only matched entities, but also equivalent properties.
– We conduct a comprehensive set of experiments to evaluate each part of our framework. Experimental results not only show that the proposed framework can generate a large-scale knowledge graph after inputting multiple online encyclopedias, but also demonstrates that our methods of live extraction and semi-supervised entity matching significantly outperform state-of-the-art baselines in different evaluation metrics.

The rest of the paper is organized as follows. Section 2 outlines the related work. Section 3 gives an overview of our proposed framework. Section 4 introduces the details of knowledge extraction, including regular extraction and live extraction. Section 5 presents our approach for knowledge linking. Section 6 shows the experimental results and finally we conclude in Section 7.

## 2 Related Work

In this section, we review the related work on knowledge extraction from online encyclopedias and knowledge linking across knowledge graphs.

### 2.1 Knowledge Extraction from Online Encyclopedias

There are many research achievements aiming to extract knowledge from the largest multilingual online encyclopedia, i.e., Wikipedia, to construct large-scale knowledge graphs. Here we introduce some typical examples. DBpedia [15] is the structured version of Wikipedia. It extracts knowledge from the information (e.g., infoboxes) in Wikipedia articles using fixed patterns. DBpedia has a live extraction system [10] which extracts knowledge from the live feed of Wikipedia, but when online encyclopedias do not provide such live feeds, this system is ineffective. YAGO [17] is a large-scale knowledge graph integrating Wikipedia, WordNet [8] and GeoNames[8]. It first designs rules to extract relations from infoboxes, and then perfroms entity type inference from article categories of each entity. After that, it maps the categories in Wikipedia to the synsets in WordNet, in order to build the ontology. YAGO also mines temporal and spatial information to generated triples. BabelNet [19] is the largest multilingual thesaurus in the world. It first maps the articles in English Wikipedia to the synsets in WordNet, and then utilizes cross-language links and machine translation to acquire the initial version of BabelNet. BabelNet further integrates OmegaWiki[9], Wiktionary[10], etc. KYLIN [34] is a self-supervised learning system which extracts knowledge from the text in English Wikipedia in a bootstrapping way with limited human guidance. Since the above work focuses on extracting knowledge from only one online encyclopedia (i.e., Wikipedia), the number of non-English knowledge (including triples, entities, etc.) is sparse. This is why we need to construct non-English knowledge graphs from multiple online encyclopedias. For example, Chinese knowledge graph construction should consider Chinese Wikipedia, Baidu Baike, Hudong Baike and etc., and Korean knowledge graph building should not neglect Doosan Encyclopedia.

There are a few publications [35] focusing on extracting knowledge from multiple online encyclopedias. Zhishi.me [24] is a Chinese knowledge graph built from three largest Chinese online encyclopedias: Baidu Baike, Hudong Baike and Chinese Wikipedia. It adopts the methods similar to those of DBpedia to extract structured knowledge from online encyclopedias and link equivalent entities across them by fixed rules. CN-DBpedia [37] is another large-scale Chinese knowledge graph, which is extracted from the same data sources with Zhishi.me. Different from Zhishi.me, CN-DBpedia fuses the extracted structured knowledge from online encyclopedias, rather than preserve different representations of equivalent knowledge from different sources. CN-DBpedia also has a live extraction system [16] used to predict the update frequency of entities in Baidu Baike. This system only extracts features at current time to make time series prediction, but our proposed live extraction utilizes a LSTM network to model the sequential input features at different time points, which has better performance in prediction. XLORE [32] is a Chinese-English bilingual knowledge graph built from semi-structured data in online encyclopedias. The focus of XLORE is to extract and link bilingual entities. Recently, XLORE2 [14] has been published which adds more facts by cross-lingual property matching and cross-lingual taxonomy alignment. Different from XLORE, our paper only concerns about monolingual knowledge

[8] `http://www.geonames.org/`
[9] `http://www.omegawiki.org/`
[10] `https://www.wiktionary.org/`

extraction and linking. A pity is that the above publications of Zhishi.me, CN-DBpedia and XLORE miss many technical details, so it is hard to reuse their construction frameworks or techniques, but our paper will clearly provide the technical details of each part in our proposed framework.

## 2.2 Knowledge Linking across Knowledge Graphs

Knowledge linking is an intense research topic in different areas, such as semantic Web, natural language processing and data mining. It includes entity matching, property matching, and etc. Here, we pay attention to the relevant work on knowledge linking across knowledge graphs.

Since lots of knowledge graphs (i.e., RDF datasets) have been published in LOD, several tools [20] such as SILK [31], KnoFuss [22], LIMES [21] have been proposed to discover links (equivalent relations) among entities from different knowledge graphs. All of these tools support manually designed match rules. Both KnoFuss and LIMES also utilize unsupervised genetic programming to generate link specifications. Besides, SILK and LIMES can incorporate supervised learning methods to improve the quality of link discovery. One extensional method of LIMES is WOMBAT [27] which performs semi-supervised learning with only positive training data, so it has a similar application scenario of our proposed semi-supervised method, and we test WOMBAT in experiments. There are also some algorithms [12,23,13] matching entities by comparing the property-value pairs. The algorithms given in [12,13] need to identify the most discriminating property-value pairs for entity matching, and how to measure the discriminability reasonably is a big issue which limits their effectiveness, but ours do not have this limitation. Our previous work [23] only outputs matched entities without equivalent properties, so we improve it in this paper by incorporating matching properties into the process of entity matching, based on the idea that we can get high-quality matched entities by using high-quality matched properties, and vice visa.

Recently, embedding techniques are used to learn continuous entity representations to match entities across knowledge graphs. MTransE [5] studies cross-lingual entity matching by spatially transforming different embedding spaces of knowledge graphs. IPTransE [39] and JAPE [28] represent different knowledge graphs into a unified embedding space by configuring parameters sharing on existing matched entities. IPTransE learns new matched entities and update embeddings in an iterative way. JAPE additionally leverages attribute embeddings to enhance entity matching. BootEA [29] is the most recent and state-of-the-art embedding-based entity matching method. It iteratively labels likely matched entities as training data for learning alignment-oriented embeddings, and employs an editing method to reduce error accumulation during iterations. We take BootEA as a baseline in our experiments.

For property matching, Zhang et al. [38] proposed statistical knowledge patterns for identifying synonymous properties in LOD. This method depends on specific data (e.g., entity types and concept properties), which do not exist

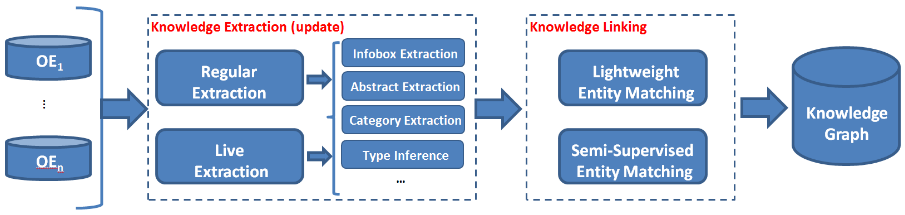


**Fig. 1** Our proposed framework (OE: Online Encyclopedia)

in every knowledge graph, so it can not always be directly applied. Rico et al. [25] presented a method to cluster properties by domain and range at first, and then identify equivalent properties using natural language processing techniques, including tokenization, stemming and lemmatization. This method also can not easily used for non-English property matching because it strongly relies on the techniques of English natural language processing.

## 3 Overview of the Framework

In this section, we present an overview of our proposed framework for knowledge graph construction from multiple online encyclopedias.

As shown in Fig. 1, we have two core modules: knowledge extraction and knowledge linking. The input of our framework are different online encyclopedias ($OE_1, OE_2, ...OE_n$). We first use regular extraction to extract different kinds of article contents as RDF triples, such as *Infobox Properties* (corresponds to infobox extraction), *Abstracts* (corresponds to abstract extraction), *Categories* (corresponds to category extraction), *Entity Types* (corresponds to type inference), and others. The interval of executing regular extractions is often long, because re-extracting all given online encyclopedias consumes too much network bandwidth and time. To update knowledge in time, we frequently utilize live extraction which relies on a supervised update frequency predictor based on LSTM, to find emerging entities and existing entities with updated article contents, and then extract their article contents as RDF triples. When we got the generated knowledge by knowledge extraction, we leverage lightweight entity matching strategies and a semi-supervised rule learning method to match entities and properties across online encyclopedias. Finally, all output RDF triples compose a knowledge graph. The above process can be optional repeated because regular extraction and live extraction are used together to support knowledge update.

## 4 Knowledge Extraction

In this section, we introduce regular extraction and live extraction in detail.

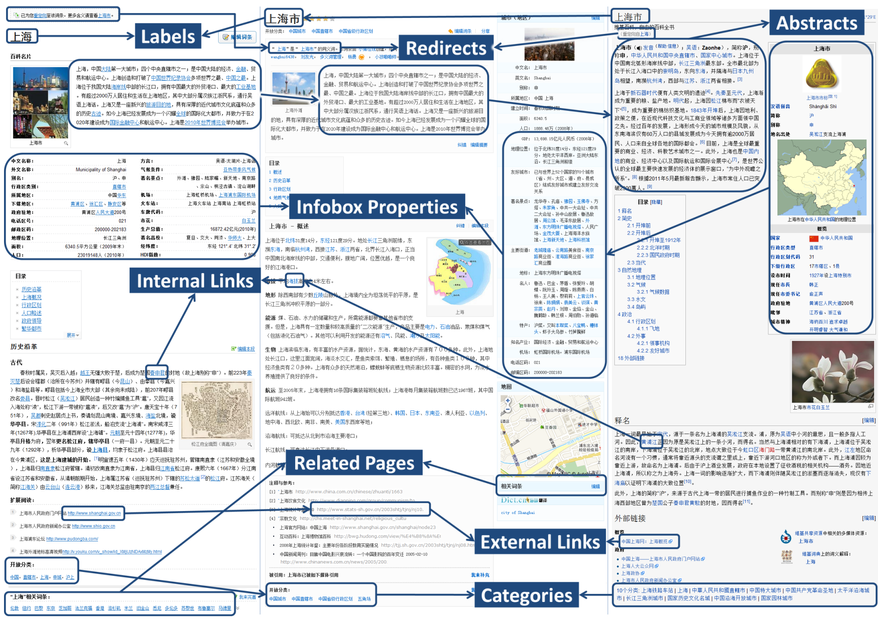


**Fig. 2** Sample article contents from Baidu Baike, Hudong Baike and Chinese Wikipedia

## 4.1 Regular Extraction

Different online encyclopedias provide different ways to edit articles and publish them. Thus, there is no one-plan-fits-all extraction strategy. Wikipedia provides database backup dumps[11], which embed all wiki articles in the form of wikitext source and meta data in XML. The techniques for extracting information from Wikipedia dumps are rather mature. The DBpedia Extraction Framework [3] is the most typical effort. Other online encyclopedias (e.g., Baidu Baike and Hudong Baike) usually provide the WYSIWYG (what you see is what you get) HTML editors. Hence, all information should be extracted from HTML file archives. Article contents come from different online encyclopedias are alike with minor differences in layout as shown in Fig. 2. Currently, we design extractors to structure different kinds of article contents, such as *Infobox Properties*, *Abstracts*, *Categories*, *Entity Types* and others. They will be explained as follows:

– **Infobox Properties.** An infobox is a table and presents some featured properties of the given article (an article corresponds to an entity, and article titles are taken as entity names).
– **Abstracts.** Online encyclopedias often have separate abstract or summary sections in articles. We extract these textual data as important descriptions of entities.

[11] `http://dumps.wikimedia.org/`

- **Categories.** Categories describe the subjects (or topics) of a given article. We also extract hyponymy relations between categories if the online encyclopedia provide a taxonomy.
- **Entity Types.** Note that categories and types are different concepts in this paper. Given an entity, its categories can be treated as candidate types, and if there exists a "*isA*" relation from a category to the entity, then the category is the type of the entity. Here, we use a general language-independent type inference method proposed by [36] to generate entity types. This method first extracts category properties and entity properties by different strategies, and then utilizes a random graph walk model to infer entity types from given categories.
- **Others.** According to the content characteristics of the chosen online encyclopedias, we can also extract some other information shown in Fig. 2, such as *Redirects*, *Internal Links*, *External Links*, *Related Pages* and etc.

## 4.2 Live Extraction

Since knowledge is not static, some of which may change (e.g., the US president is Barack Obama in 2014, but now is Donald Trump) and new knowledge is constantly emerging, we need to update the knowledge graph generated by regular extraction to prevent it from being outdated. Although we can use the proposed regular extraction to re-extract the whole online encyclopedias periodically, such a heavy-weight extraction consumes too much network bandwidth. Besides, although the Wikipedia dump is updated monthly, other online encyclopedias do not provide such dumps to download, and crawling the tens of millions of articles in other online encyclopedias usually needs more than one month with a single server, because they may ban the crawling if the access is too frequent. This is why it is very hard to update a knowledge graph frequently by regular extraction, but if the time interval between two update cycles is relatively long, it will lead to the obsolescence of knowledge. Therefore, we present a new live extraction method introduced in the following subsections to update the constructed knowledge graph in time.

### *4.2.1 Workflow*

A prerequisite for being able to perform live extraction is an access to the changes made in online encyclopedias. The Wikimedia foundation provides users with the access to the update stream of Wikipedia, i.e., the Wikipedia OAI-PMH live feed[12] which records all changes. We can pull updates in XML via HTTP, to get the Wikipedia entities with updated article contents. The DBpedia live extraction system [10] relies on this update mechanism, and we also leverage this feed service in our method to update knowledge.

[12] `http://www.mediawiki.org/wiki/Extension:OAIRepository`

However, other online encyclopedias do not provide update streams, so we should identify the entities which are newly added or have updated article contents. Actually, in a short period of time (a day or a week or even a month), we find that not too many emerging entities can be added into any online encyclopedia and only a few existing entities' article contents have been changed. Hence, we first choose the entities mentioned in the Wikipedia update streams to extract their mapped article contents in other online encyclopedias. They are either emerging entities or existing entities with updated article contents in Wikipedia. Besides, in order to improve the coverage, we also adopt an entity expansion method to get more candidate entities, and then utilize a supervised update frequency predictor based on an LSTM Network to help obtain the priority of each expanded entity for live extraction. Fig. 3 gives the whole workflow of live extraction method, which contains eight steps as follows:

1) **Stream Filtering.** Since most of the updates (text edit) in the update streams of Wikipedia are irrelevant to the knowledge we extracted, we only preserve the updates relevant to the article contents extracted in regular extraction, such as the modification of *Infobox Properties*.
2) **Entity Label Extraction.** We not only extract titles (i.e., entity labels) of articles from the filtered updates in the Wikipedia of the target language (non-English), but also those of English Wikipedia. This is because most of the updates belong to English Wikipedia when comparing with the Wikipedias of other languages. We convert such English entity labels into Chinese ones with different techniques (will be introduced in the step of Cross-Lingual Tranformation) to improve the recall of live extraction.
3) **Wikipedia Synchronizing.** With the extracted entity labels in the update streams of the Wikipedia of the target language, we extract their corresponding article contents to synchronize the knowledge graph and the Wikipedia of the target language. It is an optional step since Wikipedia is not a compulsory extraction source.
4) **Cross-Lingual Transformation.** We first use cross-lingual links between English Wikipedia and the Wikipedia of the target language to transform English entity labels into the corresponding ones. If there is no such link for a given English entity label, we then utilize machine translation[13] to get the translated one.
5) **Entity Search.** Each of the entity labels (in target language) generated by the steps of Entity Label Extraction and Cross-Lingual Transformation will be submitted as the query to the search engine of each online encyclopedia (except Wikipedia). If the search engine directly redirects to an article page when given a query, we take the query as a seed entity of the given online encyclopedia.
6) **Seed Synchronizing.** With the collected seed entities of each online encyclopedia (excluding Wikipedia), we extract their article contents for synchronization.

[13] We directly use Google Translate: `https://translate.google.com/`.

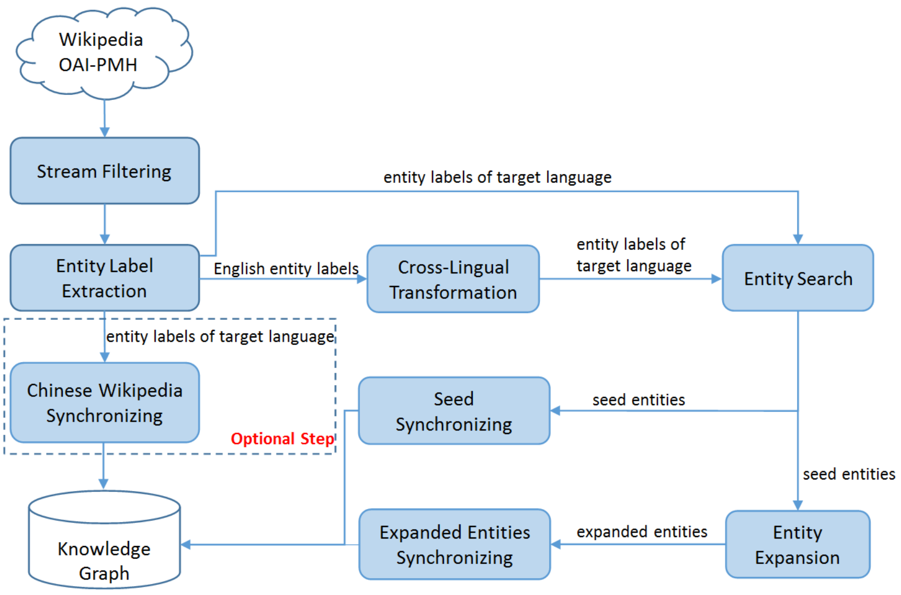


**Fig. 3** The workflow of our live extraction method

7) **Entity Expansion.** In order to find more emerging entities and existing entities with updated article contents besides the collected seed entities, we try to leverage internal links in online encyclopedias to perform entity expansion. If there exists an internal link between an entity and a seed entity, this entity will be added to the set of expanded entities. This is based on the observation that an entity that is semantically related to a recently updated entity is also likely to be updated recently.
8) **Expanded Entities Synchronizing.** For the expanded entities, we extract their article contents in order according to the criteria, which is decided by the predicted update times (denoted as $UT(e)$) of each entity $e$ after its last synchronization. The larger $UT(e)$ is, the higher priority of $e$ has. $UT(e) = F(e) \cdot D(e)$ where $F(e)$ is the predicted update frequency (will be introduced in the next subsection) and $D(e)$ is the time duration since the last synchronization of $e$. We set $D(e) = +\infty$ when $e$ is an emerging entity, so emerging entities always have the highest priority to be synchronized.

We execute the live extraction once per day, but since bandwidth resources are always limited and online encyclopedias have restricted access, we set an upper limit $K$ on the number of entities we access in one day. In the above proposed Entity Expansion (Step 7) and Expanded Entities Synchronizing (Step 8), if all article contents of the expanded entities have been extracted before the upper limit is reached, we will find more expanded entities which have internal links with existing expanded entities. In other words, these two steps are repeated until the upper access limit $K$ is reached.

### *4.2.2 Supervised Update Frequency Predictor*

We propose a supervised model to predict the update frequency. For each entity, we extract the features at the given time $t$ as follows:

1) **Number of links.** We compute the number of internal links linking to the given entity. The more such links the given entity have, the more possibly the changes of other entities might propagate to it.
2) **Number of triples.** This is the number of all triples containing the given entity. An entity existing in more triples has a larger chance to be updated since it contains rich content.
3) **Number of Infobox triples.** Infobox triples refer to the triples extracted from the infobox of the given entity's corresponding article page in an online encyclopedia. The infobox contains the factual knowledge of the entity. We use an independent feature to characterize it.
4) **Time of existence.** This is used to quantify how long an entity exists in the online encyclopedia. Intuitively, the article contents of an entity with a longer time of exisence have a larger chance to be updated.
5) **Total updates.** This quantifies how many times the article contents of an entity has been updated. If the article contents of an entity have been updated many times in the past, then the entity's article contents may have a higher probability to be updated in the future.
6) **Historical update frequency.** This is the historical average update frequency since the given entity has been created.
7) **Short-term update frequency.** This is the average update frequency for the given entity in the past month.

In addition, for feature 1-5, we use $g(x) = log(1 + x)$ ($x$ denotes the feature value) to rescale the value of each feature.

A good update frequency predictor should not only well model our proposed features for each entity, but also capture change tendencies of these features. Thus, we collect the features of each entity at different time points as the training data which are fed into the core module of our proposed supervised update frequency predictor, i.e., a Long Short-Term Memory (LSTM) Network [11]. The LSTM Network is a kind of Recurrent Neural Network [18], which has been proved being powerful in handling sequence problems. We show the whole structure of our supervised update frequency predictor in Fig. 4. For a given entity $e$, the input of the LSTM Network is $k+1$ feature vectors $\mathbf{X}_1^{t^-}(e)$, $\mathbf{X}_2^{t^-}(e)$, ..., $\mathbf{X}_k^{t^-}(e)$ and $\mathbf{X}^t(e)$, in which $\mathbf{X}^t(e)$ is the feature vector of $e$ calculated based on the information at current time $t$, and $\mathbf{X}_1^{t^-}(e)$, $\mathbf{X}_2^{t^-}(e)$, ..., $\mathbf{X}_k^{t^-}(e)$ are $k$ feature vectors calculated at $k$ time points respectively before time $t$. Here, to obtain these $k$ feature vectors of each entity, we choose to extract its features once a week as a fixed extraction. Besides, when we need to predict the update frequency of an entity in the past, we also extract its features. Based on such extractions at different time points, we get and only preserve the most recent $k$ feature vectors of each entity before current time $t$. The output of the LSTM Network is $\langle \mathbf{y}_1^{t^-}, \mathbf{y}_2^{t^-}, ..., \mathbf{y}_k^{t^-}, \mathbf{y}^t \rangle$, which is a concate-

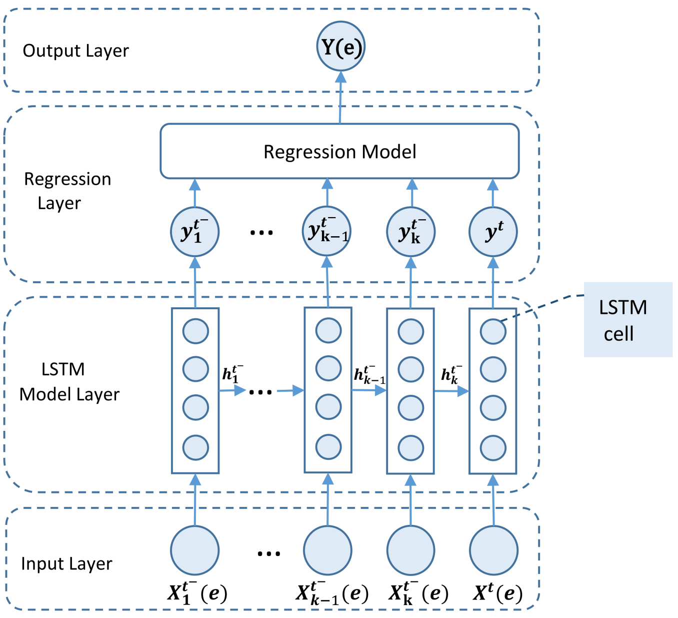


**Fig. 4** The structure of our proposed supervised update frequency predictor

nated vector used as the input of a regression model (e.g., linear regression or random forest regression). $\mathbf{y}^t = \sigma(W\mathbf{h}^t + \mathbf{b})$, where $\sigma$ is the sigmoid function, $W$ is a weight matrix, $\mathbf{b}$ is a bias vector, and $\mathbf{h}^t$ is the state of the hidden layer at current time $t$ and it is also the output vector of gate operations in LSTM cells. $\mathbf{y}_1^{t^-}$, $\mathbf{y}_2^{t^-}$, ..., $\mathbf{y}_k^{t^-}$ are computed in the same way. Finally, the output of the regression model $Y(e)$ is the predicted update frequency of $e$ at time $t$.

To train the predictor, we apply the algorithm of Backpropagation Through Time [33] and use the mean square error loss function as the objective function as follows:

$$L = \frac{1}{|E_{train}|} \sum_{e \in E_{train}} (Y_g(e) - Y(e))^2 \tag{1}$$

where $Y_g(e)$ is the ground-truth (i.e., the actual update frequency) of entity $e$, $|E_{train}|$ denotes the number of all elements in the set $E_{train}$, which contains all entities in the training data.

## 5 Knowledge Linking

Since we study knowledge graph construction from multiple online encyclopedias, which contain duplicated entities and their descriptions are similar and complementary, we expect to link equivalent entities across different online encyclopedias to tackle the heterogeneity issue. In this section, we introduce our approach for automatical entity matching, which contains two parts: 1) the lightweight entity matching using three simple strategies, and 2) the semi-supervised entity matching which takes the lightweight entity matching results as seeds to find more equivalent entities. In addition, since the proposed semi-supervised method iteratively learns equivalent properties to help construct entity matching rules, we can not only harvest equivalent entities but also matched properties.

## 5.1 Lightweight Entity Matching

In this part, we use three lightweight strategies which are *Using Entity Labels*, *Punctuation Cleaning* and *Extending Synonyms*, to acquire high-quality matched entities. These strategies are introduced as follows.

1) **Using Entity Labels.** This strategy directly compares the labels of two given entities, if the labels are exactly the same, then the entities are equivalent. It normally has a high precision except it comes with the problem of homonyms. Fortunately, to distinguish the homonyms as different entities, online encyclopedias usually use different labels to denote them, such as "*Li Na* and *Li Na (diver)* in Wikipedia". In other words, it is impossible to find two entities in an online encyclopedia that have different meanings with the same label. This fact ensures the correctness of this strategy.
2) **Punctuation Cleaning.** Sometimes, equivalent entities from different online encyclopedias have different labels due to the different usage of punctuation marks. If the labels of two given entities are exactly the same after removing some specific punctuation marks, then the entities are equivalent. We use some Chinese examples in Fig. 5 to illustrate why punctuation cleaning is effective. In the first example, editors may use guillemets (≪≫) to indicate the title of a book, film or album etc. in some Chinese online encyclopedias, but guillemets are not imperative to be part of titles. In the second example, Chinese people often insert an interpunct (·) or a hyphen between two personal name components. In this situation, the interpunct or hyphen can be removed. In the last example, it is always a good practice in Chinese to quote a cited name by double styling quotation marks (“”), so they can also be removed in our punctuation cleaning strategy. Other language may have other rules of punctuation cleaning, but the idea of this strategy is general.

(1) 肖申克的救赎 = 《肖申克的救赎》 **(The Shawshank Redemption)**
(2) 海尔波普彗星 = 海尔·波普彗星 = 海尔-波普彗星 **(Comet Hale-Bopp)**
(3) 奋进号航天飞机 = “奋进号” 航天飞机 **(Space Shuttle Endeavour)**

**Fig. 5** Examples of equivalent entities with different labels

3) **Extending Synonyms.** In this strategy, we first collect high-quality synonym relations (i.e., `owl:sameAs`[14]) obtained from the *Redirects* information (A redirects to B means A and B are synonyms). We then extend the synonymous entity labels by the transitive property of `owl:sameAs`, i.e., if A `owl:sameAs` B, B `owl:sameAs` C, then A `owl:sameAs` C. Based on such synonymous entity labels, if the labels of two given entities from different online encyclopedias are synonyms, then the entities are equivalent.

## 5.2 Semi-Supervised Entity Matching

Although the lightweight entity matching strategies could generate high-quality equivalent entities by mapping label variants, lots of structured information (e.g., *Infobox Properties*, *Categories*, etc.) is not leveraged. We believe that more implicit equivalent entities can be mined from different online encyclopedias when using such information. Thus, we propose a semi-supervised method to learn property-based rules for entity matching. At first, we give a typical example to illustrate what a property-based entity matching rule is. Suppose we have three pairs of equivalent properties from some known matched entities (can be obtained from lightweight entity matching) as follows (in Table 1):

**Table 1** Examples of equivalent properties

| | | |
|---|---|---|
| `baidu:` 中文名 (Chinese name) ($p_{11}$) | $\approx$ | ($p_{21}$) `hudong:` 译名 (translated name) |
| `baidu:` 导演 (director) ($p_{12}$) | $\approx$ | ($p_{22}$) `hudong:` 导演 (director) |
| `baidu:` 上映日期 (release date) ($p_{13}$) | $\approx$ | ($p_{23}$) `hudong:` 首映日期 (release date) |

where $p_{ij}$ denotes the $j$th property extracted from the $i$th source, "`baidu:`" and "`hudong:`" mean the given properties are extracted from Baidu Baike and Hudong Baike, respectively. Based on these pairs of equivalent properties and known matched entities, we may learn a rule like this: if two entities $e_1$ and $e_2$ satisfy

$$\bigwedge_{j=1}^{3} \big(p_{1j}(e_1, o_1) \wedge p_{2j}(e_2, o_2) \wedge o_1 \simeq o_2\big)$$

($p(e, o)$ is the function expression of the triple $\langle e, p, o\rangle$, and $o_1 \simeq o_2$ means both property values $o_1$ and $o_2$ refer to the same entity or literal), then $e_1$ and $e_2$ are equivalent.

Fig. 6 shows the workflow of the proposed semi-supervised method. The input of the *Generator* is known matched entities (initially are the results of lightweight entity matching) with their corresponding property-value pairs. The *Generator* outputs frequent sets of property pairs using association rule mining. Then, the *Rule Constructor* uses such frequent sets of property pairs,

[14] `owl:sameAs` denotes the equality relation (between individual entities) defined by the W3C Web Ontology Language.

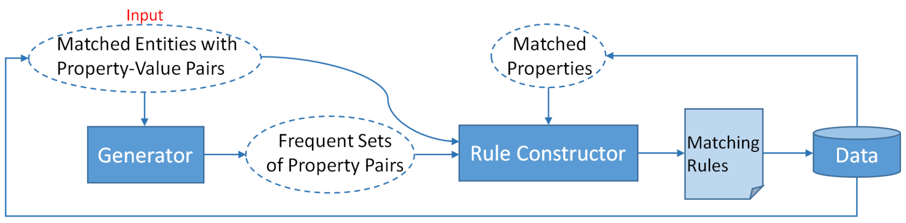

Input
Matched Entities with Property-Value Pairs
Generator
Frequent Sets of Property Pairs
Matched Properties
Rule Constructor
Matching Rules
Data

**Fig. 6** The workflow of our semi-supervised entity matching

property-value pairs of matched entities and existing matched properties (initially we do not have matched properties) to produce entity matching rules. After applying these rules to unmatched entities, we can get new matches (i.e., matched entities and matched properties) with high confidence. We execute the above process in an iterative way based on the Expectation-Maximization algorithm [6] until no matched entities are found in a certain iteration.

Before introducing our method in detail, we first introduce some basic concepts and definitions. An entity that refers to a real-word thing is usually described by several property-value pairs (described in *Infobox Properties*). A property-value pair along with an entity constitute a triple $\langle s, p, o \rangle$. Generally, the subject ($s$) is the entity mentioned above, the predicate ($p$) stands for the property and the object ($o$) stands for the property value, which can be another entity or a literal. The triples extracted from one online encyclopedia constitute a graph $G$.

**Definition 1** *(Equivalence of Entities) Equivalence of entities, denoted by $\sim_{\mathcal{I}}$, is an equivalence relation. It indicates that two entities are actually the same thing in the world. Two entities, $e_1$ and $e_2$, which are extracted from different sources are equivalent iff. $(e_1, e_2) \in \sim_{\mathcal{I}}$.*

**Definition 2** *(Equivalence of Properties) Equivalence of properties, denoted by $\sim_{\mathcal{P}}$, is an equivalence relation. It indicates that two properties actually have the same intensional meaning. Two properties, $p_1$ and $p_2$, are equivalent iff. $(p_1, p_2) \in \sim_{\mathcal{P}}$.*

**Definition 3** *(Equivalence of Property-Value Pairs) Equivalence of property-value pairs, denoted by $\sim_{\mathcal{PV}}$, is an equivalence relation. Given two property-value pairs ($\langle p_1, o_1 \rangle$ and $\langle p_2, o_2 \rangle$), the two pairs are equivalent iff. $(p_1, p_2) \in \sim_{\mathcal{P}}$ , $(o_1, o_2) \in \sim_{\mathcal{I}}$ (if $o_1, o_2$ are both entities) or $o_1 = o_2$ (if $o_1, o_2$ are both literals).*

In Definition 3, when $o_1, o_2$ are both string literals, we consider them as equal only when are exactly the same. Besides, for the literals of specific data types, such as distance, date, weight, etc., we use a normalization module to decide whether two given literals are equivalent. For example, the values "*1.5 km*" and "*1500 m*" of the property *distance* of two entities should be recognized as equivalents.

Now we extend the equivalence relation from two property-value pairs to property-value pair sets. Given an entity $e$ and a set of properties $P$, we have a property-value pair set $\mathrm{PV}_{e,P} = \{\langle p, o \rangle | p \in P, \langle e, p, o \rangle \in G\}$.

**Definition 4** *(Equivalence of Property-Value Pair Sets) $\sim_{\mathcal{S}}$ is an equivalence relation used to denote equivalence of property-value pair sets. Given an entity $e_1$ with its set of properties $P_1$, and another entity $e_2$ with its set of properties $P_2$ ($P_1$ and $P_2$ have the same size), two property-value pair sets ($PV_{e_1,P_1}$ and $PV_{e_2,P_2}$) are equivalent iff. there exists a bijection $f$ from $PV_{e_1,P_1}$ to $PV_{e_2,P_2}$ and $f \in \sim_{\mathcal{PV}}$.*

**Definition 5** *(Entity Matching Rule) Given two matched entities $e_1^\star$, $e_2^\star$, and two sets (same size) of properties $P_1$, $P_2$, if $(PV_{e_1^\star,P_1}, PV_{e_2^\star,P_2}) \in \sim_{\mathcal{S}}$, then we construct a set of pairs of equivalent properties $eps = \{\langle p_{1j}, p_{2j}\rangle\}_{j=1}^{|eps|}$, $p_{1j} \in P_1$, $p_{2j} \in P_2$, $|eps| = |P_1| = |P_2|$, and define an entity matching rule as*

$$\forall e_1 \forall e_2 . \left( \bigwedge_{j=1}^{|\text{eps}|} \Big( \exists o_1 \exists o_2 . \big( p_{1j}(e_1, o_1) \wedge p_{2j}(e_2, o_2) \wedge \sim_{\mathcal{PV}} (\langle p_{1j}, o_1\rangle, \langle p_{2j}, o_2\rangle)\big)\Big) \rightarrow \sim_{\mathcal{I}} (e_1, e_2) \right) \tag{2}$$

In Definition 5, $e_1$ and $e_2$ represent entities, $o_1$ and $o_2$ are property values, and $p(e, o)$ is the function expression of the triple $\langle e, p, o\rangle$. Since a property may have more than one value, requesting all values have equivalents are too strict. This is why we use existential quantification $\exists o_1 \exists o_2$. here.

### *5.2.1 The Expectation-Maximization Algorithm*

The procedure of discovering rules as well as matched entities and matched properties in an iterative way follows the Expectation-Maximization (EM) algorithm. The EM algorithm is an iterative procedure to estimate missing data by estimating the model parameter(s) for which the observed data are the most likely. The EM iteration alternates between performing an expectation step (E-step), and a maximization step (M-step). The E-step estimates the missing data (i.e., matched entities and matched properties) using the observed data and the current estimate for the parameters. In our scenario, we regard the entity matching rules, denoted by $\boldsymbol{\theta}$, as the model parameters. The M-step computes parameters by maximizing the likelihood function. The data estimated in E-step are used in replace of the actual missing data.

Before giving the definition of our specific likelihood function used to compute model parameters, we first introduce the concept of Entity Matching Graph (EMGraph).

**Definition 6** *(Entity Matching Graph (EMGraph)) A EMGraph is an undirected graph, in which a vertex represents an entity and an edge links two vertices if their corresponding entities are supposed to be equivalent. A unique EMGraph can be obtained by applying a set of entity matching rules to unmatched entities.*

The likelihood function is considered to be a function of $\boldsymbol{\theta}$ given the EMGraph $\boldsymbol{M}$, so we define it as

$$L(\boldsymbol{\theta}; \boldsymbol{M}) = \Pr(\boldsymbol{M}|\boldsymbol{\theta}) \tag{3}$$

Naturally, the probability of $\boldsymbol{M}$ is reflected in the proximity of $\boldsymbol{M}$ and the EMGraph built by actual matched entities. The proximity is usually reflected in two aspects: correctness and completeness. Precision and recall are two widely used metrics to evaluate these two characteristics respectively [7]. However, without complete reference matched entities, it is difficult to evaluate either of them. Hence we propose an alternative measurement, that is optimizing the precision takes priority (ensuring the precision is higher than a given threshold) and obtaining all potential matched entities on the premise of that precision value. Thus, Equation (3) can be continued as,

$$L(\boldsymbol{\theta}; \boldsymbol{M}) \approx \text{Precision}(\boldsymbol{M}|\boldsymbol{\theta}) \tag{4}$$

For the EMGraph $\boldsymbol{M}$, we can estimate its approximate precision value by evaluating the divergence of this graph:

$$\begin{aligned} \text{Precision}(\boldsymbol{M}|\boldsymbol{\theta}) \approx & \text{Divergence}(\boldsymbol{M}) \\ = & \frac{|\text{ConnectedComponent}(\boldsymbol{M})|}{|\text{Edge}(\boldsymbol{M})|} \end{aligned} \tag{5}$$

where $|\text{ConnectedComponent}(\boldsymbol{M})|$ is the number of connected subgraphs in $\boldsymbol{M}$, and $|\text{Edge}(\boldsymbol{M})|$ is the number of edges in $\boldsymbol{M}$. Since entities (vertices) in $\boldsymbol{M}$ come from two data sources and assuming that no equivalent entities exist in a single data source, we can infer that an entity is equivalent to at most one other from the other data source and thus $\text{Divergence}(\boldsymbol{M})$ should be 1. Incorrect matched entities in $\boldsymbol{M}$ may result in a vertex connecting to more than one other vertices, which is contrary to the assumption, and decrease the divergence of $\boldsymbol{M}$.

For each parameter $\theta$ (i.e., an entity matching rule), we construct an $M$, and put it into the final united $\boldsymbol{M}$. To maximize $\text{Precision}(\boldsymbol{M}|\boldsymbol{\theta})$, setting a high threshold for each $\text{Precision}(M|\theta)$ is our chosen solution. Once the threshold is determined, the number of $M$s, as well as the entity matching rules are also determined. Note that each rule takes the corresponding $\text{Divergence}(M)$ as its confidence value.

Algorithm 1 demonstrates the overall framework of the implementation of our EM algorithm. The input is the set of triples extract from two sources, and a set of pairs of known matched entities `KMEset` obtained from lightweight entity matching. We first initialize a set of pairs of known matched properties `KMPset` as $\emptyset$ (*line 1*). Then, we iteratively execute the M-step (*line 3*) and E-step (*line 4*) to generate a set of rule-confidence pairs `RULE_CONF_set`, a set of pairs of matched entities `MEset` and a set of pairs of matched properties `MPset` until no new matched entities are found (i.e., `new_MEset` $= \emptyset$) in some iteration (*line 8*). Finally, we output updated `KMEset` and `KMPset`. The details of the E-step and M-step are introduced in the next subsections.

**Algorithm 1**: The Expectation-Maximization Algorithm

```
  Input: a set of triples Tset, a set of pairs of known matched entities KMEset.
  Output: updated KMEset, a set of pairs of matched properties KMPset.
1 KMPset ← $\emptyset$;
2 repeat
3     RULE_CONF_set ← $minRules$(Tset, KMEset, KMPset);
4     MEset, MPset ← $getMatches$(Tset, RULE_CONF_set);
5     new_MEset = MEset − KMEset;
6     KMEset = KMEset ∪ new_MEset;
7     KMPset = KMPset ∪ MPset;
8 until new_MEset = $\emptyset$ ;
9 return KMEset, KMPset;
```

### *5.2.2 M-step: Mining Entity Matching Rules*

The M-step mines entity matching rules by maximizing the likelihood function (Equation (4)) given the set of triples `Tset`, the set of pairs of known matched entities `KMEset`, and the set of pairs of known matched properties `KMPset`. Algorithm 2 presents the details of the M-step.

For each pair (i.e., $c$) of matched entities, the triples containing their property-value pairs are merged to form a transaction (*line 2-3*). Then, we mine a set of pairs of candidate matched properties `candidate_MPset` from the set of such transactions `trans` by association rule mining (*line 4*), which only uses the properties in transactions. We adopt the method proposed by Völker et al. [30] for association rule mining, which acquires property subsumption axioms by mining binary association rules, so an equivalence relation can be denoted by subsumption relations on both sides of a given property pair. In practice, some weakly related properties that have overlaps in semantics are also considered to be involved in building rules. We set a relatively low threshold 0.1, to output more candidate matched properties. After that, we mine frequent sets of property pairs `frequent_PPsets` from `candidate_MPset` using the classic Apriori algorithm [1] (*line 5*). For each frequent set $P \in$ `frequent_PPsets` consisting of one or more pairs of candidate matched properties, we construct two subsets, $P_1$ and $P_2$. In all property pairs in $P$, all properties from one source and another source compose $P_1$ and $P_2$, respectively.

Given a pair of matched entities $c = \langle e_1, e_2 \rangle$ in a transaction $t$ and a frequent set of candidate matched properties $P$, if the property-value sets (i.e., $PV_{e_1^{ct},P_1}$ and $PV_{e_2^{ct},P_2}$) are equivalent based on Definition 4, then $P$ will be added to the set `PPRsets` (*line 7-10*) which might be used to construct entity matching rules later. To obtain more high-quality rules, we try to use the set of known matched properties `KMPset` discovered from the previous iterations to enrich `PPRsets` (*line 11-17*). We first generate the set of all possible property pairs `APPTset` from $t$. For each property pair $pp \in$ `APPTset`, if $pp \in$ `KMPset` and the properties in $pp$ do not exist in any property pair in $P$, we add $pp$ into the set `sub_KMPset`. Then, we add each subset `set` $\subset$ `sub_KMPset` combining with $P$ into `PPRsets`.

**Algorithm 2**: Mining Entity Matching Rules (*mineRules*)

```
Input: a set of triples Tset, a set of pairs of known matched entities KMEset, and a
       set of pairs of known matched properties KMPset.
Output: a set of rule-confidence pairs RULE_CONF_set.
1  trans ← $\emptyset$;
2  foreach $c \in$ KMEset do
3      trans ← trans $\cup \{\{\langle s,p,o\rangle | \langle s,p,o\rangle \in$ Tset$, s \in \{e_1^c, e_2^c\}\}\}$;
4  candidate_MPset ← $mineAssociationRules$(trans);
5  frequent_PPsets ← $mineFrequentSets$(candidate_MPset);
6  PPRsets ← $\emptyset$;
7  foreach $t \in$ trans do
8      foreach $P \in$ frequent_PPsets do
9          if $(\mathrm{PV}_{e_1^{c^t}, P_1}, \mathrm{PV}_{e_2^{c^t}, P_2}) \in \sim_{\mathcal{S}}$ then
10             PPRsets ← PPRsets $\cup \{P\}$;
11             generate a set of all possible property pairs APPTset from $t$;
12             sub_KMPset ← $\emptyset$;
13             foreach $pp \in$ APPTset do
14                 if $pp \in$ KMPset and the properties in $pp$ do not exist in any
                   property pair in $P$ then
15                     sub_KMPset ← sub_KMPset $\cup \{pp\}$;
16             foreach $set \subset sub\_KMPset$ do
17                 PPRsets ← PPRsets $\cup \{P \cup$ set$\}$;
18 RULE_CONF_set ← $\emptyset$;
19 foreach $P \in$ PPRsets do
20     if Divergence$(M_{rule(P)}) > \epsilon_r$ and Divergence$(M_{rule(P)}) >$
21     all Divergence$(M_{rule(P^\star)})$ where $P^\star \subset P$ and $P^\star \in$ PPRsets then
22         RULE_CONF_set ← RULE_CONF_set $\cup \{\langle rule(P), \mathrm{Divergence}(M_{rule(P)})\rangle\}$;
23 return RULE_CONF_set;
```

The final part is to generate the set of high-quality rules (*line 18-22*). For each $P \in$ `PPRsets` and each subset $P^\star \subset P$, we construct entity matching rules $rule(P)$ and all $rule(P^\star)$ based on Definition 5. After applying these rules to construct EMGraphs: $M_{rule(P)}$ and all $M_{rule(P^\star)}$, we maximize the likelihood function (Equation (5)) by setting a high threshold $\epsilon_r = 0.9$, and add another constraint that the confidence value (i.e., likelihood value) of the given rule $rule(P)$ should be larger than that of any sub-rule $rule(P^\star)$ of itself.

### *5.2.3 E-step: Getting Matches*

The E-step estimates matches (i.e., matched entities and matched properties) with high confidence using the current estimate for the entity matching rules. Algorithm 3 presents the details of the E-step.

We first acquire a set of pairs of candidate matched entities `cand_MEset` by applying the rules in the given `RULE_CONF_set` to a set of triples `Tset` (*line 1*), and then extract all property pairs in each rule in `RULE_CONF_set` to compose a set of candidate matched properties `cand_MPset` (*line 2*). For each

**Algorithm 3**: Getting Matches (getMatches)

```
   Input: a set of triples Tset, a set of rule-confidence pairs RULE_CONF_set.
   Output: a set of pairs of matched entities MEset, a set of matched properties MPset.
1  obtain a set of pairs of candidate matched entities cand_MEset by applying the rules
   in RULE_CONF_set to Tset;
2  extract all property pairs in each rule in RULE_CONF_set to compose a set of
   candidate matched properties cand_MPset;
3  MEset, MPset ← $\emptyset$;
4  foreach $c \in$ cand_MEset do
5      if $Econf(c) > \epsilon_e$ then
6          MEset ← MEset $\cup \{c\}$;
7  foreach $pp \in$ cand_MPset do
8      if $Pconf(pp) > \epsilon_p$ then
9          MPset ← MPset $\cup \{pp\}$;
10 return MEset, MPset;
```

entity pair $c \in$ `cand_MEset`, if its confidence value $Econf(c)$ is larger than the threshold $\epsilon_e = 0.99$, we will add $c$ into the set of pairs of matched entities `MEset` (*line 4-6*). To compute $Econf(c)$, we need to consider the confidences of all rules (composing a set `RULEset(c)` $= \{rule_k^c\}_{k=1}^{|\texttt{RULEset(c)}|}$ extracted from `RULE_CONF_set`) which can acquire $c$. Intuitively, entity matching rules can been regarded as evidences and the more evidences we have, the more likely true the conclusion is. If $|\texttt{RULEset(c)}| = 1$, then we set $Econf(c)$ as the confidence of the only one rule in `RULEset(c)`. Otherwise, we iteratively combine the confidences of all rules in `RULEset(c)` using the Dempster's rule[15], which is based on the Dempster-Shafer theory, a mathematical theory of evidence [26]. In the first iteration, we combine the confidences of two rules $conf(rule_1^c)$ and $conf(rule_2^c)$ (also obtained from `RULE_CONF_set`) using the following formula:

$$conf(rule_1^c) \oplus conf(rule_2^c) = \frac{conf(rule_1^c) \cdot conf(rule_2^c)}{1 - conf(rule_1^c) - conf(rule_2^c) + 2 \cdot conf(rule_1^c) \cdot conf(rule_2^c)} \quad (6)$$

In each of the next iterations, we combine the result of the last iteration and the confidence of another rule in `RULEset(c)` using the same formula. Finally, we set $Econf(c)$ as the combined confidence.

Similarly, for each property pair $pp \in$ `cand_MPset`, if its confidence value $Pconf(pp)$ is larger than the threshold $\epsilon_p = 0.96$, we will add $pp$ into the set of pairs of matched properties `MPset` (*line 7-9*). To compute $Pconf(pp)$, we also need to consider the confidences of all rules (composing a set `RULEset(pp)` $= \{rule_l^{pp}\}_{l=1}^{|\texttt{RULEset(pp)}|}$ from `RULE_CONF_set`) containing $pp$. If $|\texttt{RULEset(pp)}| = 1$, we can only get the confidence of one rule in `RULEset(pp)`. Otherwise, we also use the Dempster's rule to combine the confidences of all rules in

[15] The Dempster's rule has the best performance in combining entity matching rules in our previous work [23] when comparing with other combination methods.

`RULEset(pp)`. Actually, there are some weakly related properties in the rules output by the M-step, and we observe that such properties exist in only a few rules for discovering specific matched entities, so we argue that if more high-quality rules contain *pp*, the probability of *pp* being a pair of matched properties is higher. Hence, we obtain $Pconf(pp)$ by multiplying the combined confidence iteratively computed by Equation (6) and a designed coefficient $1/(1+\exp^{-|\texttt{RULEset(pp)}|})$.

## 6 Experiments

In this section, we evaluated our framework from two perspectives: 1) the evaluation of the approaches for knowledge extraction and linking to show their superiority when comparing with state-of-the-art baselines; 2) the evaluation of the knowledge generated by applying our framework to real-world online encyclopedias to demonstrate the framework can generate a large-scale knowledge graph.

### 6.1 Evaluation of Methods

Since our proposed regular extraction and lightweight entity matching are simple and intuitive engineering solutions without research-based models or algorithms, it is easy to understand that they can acquire good results as long as codes are correct. Due to this fact, we focus on evaluating our proposed new methods of live extraction and semi-supervised entity matching in this subsection.

#### *6.1.1 Evaluation of Live Extraction*

We evaluated our live extraction method on two datasets collected from Chinese online encyclopedias: Baidu Baike and Hudong Baike, respectively. To build an update frequency predictor for the entities extracted from each online encyclopedia, we first computed 5 groups (i.e., we set $k = 4$ in the LSTM network) of our proposed features for each entity in these two online encyclopedias at different time points. Such a computation relies on the continuously extraction of the articles in Baidu Baike and Hudong Baike from June to August in 2018. Then, we randomly selected 100,000 entities from each online encyclopedia, and computed the average of weekly update frequency for each entity in September, 2018. After obtaining the labeled data, for each group of labeled entities, we used all of them to model training.

With the trained update frequency predictors, how to test their effectiveness in real-world applications is introduced as follows. We first collected the expanded entities for each online encyclopedia in three randomly selected days (i.e., three groups of expanded entities) from October 1, 2018 to October 15, 2018, and then monitored whether these expanded entities were updated in

**Table 2** Details of the collected expanded entities in three randomly selected days

| Online Encyclopedias | Baidu Baike | | | Hudong Baike | | |
|---|---|---|---|---|---|---|
| | day 1 | day 2 | day 3 | day 1 | day 2 | day 3 |
| #Expanded Entities | 881 | 850 | 908 | 877 | 916 | 931 |
| #Updated Expanded Entities | 358 | 321 | 373 | 298 | 308 | 313 |

**Table 3** Overall comparison results on Baidu Baike

| Metric | **Random** | **Liang** | **LSTM+LR** | **LSTM+RFR** |
|---|---|---|---|---|
| MAP | 0.261 | 0.550 | 0.607 | **0.643** |
| Precision@20 | 0.267 | 0.883 | 0.867 | **0.917** |
| Recall@20 | 0.021 | 0.078 | 0.087 | **0.137** |
| F1@20 | 0.039 | 0.144 | 0.158 | **0.234** |
| Precision@100 | 0.237 | 0.650 | 0.693 | **0.783** |
| Recall@100 | 0.101 | 0.293 | 0.323 | **0.360** |
| F1@100 | 0.141 | 0.400 | 0.439 | **0.493** |

**Table 4** Overall comparison results on Hudong Baike

| Metric | **Random** | **Liang** | **LSTM+LR** | **LSTM+RFR** |
|---|---|---|---|---|
| MAP | 0.231 | 0.506 | 0.540 | **0.590** |
| Precision@20 | 0.200 | 0.717 | 0.750 | **0.783** |
| Recall@20 | 0.024 | 0.074 | 0.075 | **0.147** |
| F1@20 | 0.042 | 0.133 | 0.137 | **0.246** |
| Precision@100 | 0.217 | 0.55 | 0.617 | **0.690** |
| Recall@100 | 0.092 | 0.275 | 0.280 | **0.303** |
| F1@100 | 0.128 | 0.359 | 0.382 | **0.421** |

the next month after each selected day (the upper limit was set to 1,000 on the number of entities we can access to an online encyclopedia per day). The details are given in Table 2.

In each group of the expanded entities (in Table 2), we expected that those updated expanded entities can be ranked at the top when using our model. We tested linear regression and random forest regression in our proposed LSTM network, which are denoted as **LSTM+LR** and **LSTM+RFR**, respectively. We also compared our model with another two methods: 1) random ordering of the expanded entities (denoted as **Random**) and 2) also a random forest regression model proposed by Liang et al. [16] with their own proposed features (denoted as **Liang**). The evaluation metrics used here are Mean Average Precision (MAP), Presion@K (P@K), Recall@K (R@K) and F1@K. The overall comparison results on Baidu Baike and Hudong Baike are given in Table 3 and Table 4, respectively. We can see that our proposed model **LSTM+RFR** has the best performance in all evaluation metrics on both online encyclopedias, and **LSTM+LR** outperforms **Random** as well as **Liang** in almost all evaluation metrics. This not only reflects the effectiveness of our LSTM model in update frequency prediction, but also shows that using random forest regression is better than using linear regression, which may indicate that non-linear regression is more suitable in this scenario. Although the state-of-the-art method **Liang** and our model have a similar set of features, **Liang** only

**Table 5** Real-world update performance of our live extraction system

| **Baidu Baike** | | | |
|---|---|---|---|
| Entities from Stream | | Expanded Entities | |
| average number | average success ratio | average number | average success ratio |
| 107 | 0.710 | 893 | 0.540 |
| **Hudong Baike** | | | |
| Entities from Stream | | Expanded Entities | |
| average number | average success ratio | average number | average success ratio |
| 85 | 0.576 | 915 | 0.479 |

inputs the features at current time, so it cannot capture the change tendencies of the features, but our model can overcome this problem because the recurrent neural network (e.g., the LSTM network) has advantages in modeling time series.

We also tested the performance of the whole live extraction method. We computed the average ratio of finding the entities which do have updated article contents each day (upper access limit = 1,000). Table 5 gives the statistics, which is based on the data collected from November 1 to November 7 in 2018. The average success ratio of top-ranked entities from each online encyclopedia is around 0.5 which demonstrates that our method is effective to keep the knowledge up-to-date, without incurring overloads on the website of online encyclopedias. We also find that the success ratio of the entities extracted from Wikipedia update streams is higher than that of expanded entities. This shows that the updates in different online encyclopedias are relatively consistent at the same time range.

### *6.1.2 Evaluation of Semi-Supervised Entity Matching*

To test the effectiveness of our proposed method of semi-supervised entity matching, we compared it with the state-of-the-art embedding-based method BootEA [29] on the datasets collected from Baidu Baike, Hudong Baike and Chinese Wikipedia. For each pair of these online encyclopedias, we randomly selected 60,000 entity matches (from the results from lightweight entity matching) as positive data and generated $60,000 \times 60,000 - 60,000$ entity pairs (from the Cartesian product except the selected entity matches) as negative samples. The above data were used for testing. Additionally, we chose different number of pairs of known matched entities (from the results of lightweight entity matching) as the labeled data in training. Note that each entity should have infobox properties. Evaluation metrics are precision (P), recall (R), and F1-score (F1). The comparison results on different pairs of online encyclopedias are shown in Table 6, Table 7 and Table 8, respectively, and we can see that:

– When given different number of labeled data in training, our method outperforms BootEA in F1-score on all test datasets corresponding to different pairs of online encyclopedias, which shows the value of our proposed semi-supervised rule learning method to entity matching across real-world online encyclopedias.

**Table 6** Comparison results of the semi-supervised entity matching between Baidu Baike and Hudong Baike

| | Baidu Baike ⟷ Hudong Baike | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | #labeled data: 20,000 | | | #labeled data: 40,000 | | | #labeled data: 60,000 | | |
| Method | P | R | F1 | P | R | F1 | P | R | F1 |
| Ours | 0.878 | 0.133 | 0.231 | 0.895 | 0.202 | 0.330 | **0.910** | 0.246 | **0.387** |
| BootEA | 0.164 | 0.180 | 0.172 | 0.212 | 0.235 | 0.223 | 0.231 | **0.250** | 0.240 |

**Table 7** Comparison results of the semi-supervised entity matching between Baidu Baike and Chinese Wikipedia

| | Baidu Baike ⟷ Chinese Wikipedia | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | #labeled data: 20,000 | | | #labeled data: 40,000 | | | #labeled data: 60,000 | | |
| Method | P | R | F1 | P | R | F1 | P | R | F1 |
| Ours | 0.896 | 0.090 | 0.164 | **0.909** | 0.159 | 0.271 | 0.906 | **0.228** | **0.364** |
| BootEA | 0.180 | 0.073 | 0.104 | 0.191 | 0.095 | 0.127 | 0.200 | 0.097 | 0.131 |

**Table 8** Comparison results of the semi-supervised entity matching between Hudong Baike and Chinese Wikipedia

| | Hudong Baike ⟷ Chinese Wikipedia | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | #labeled data: 20,000 | | | #labeled data: 40,000 | | | #labeled data: 60,000 | | |
| Method | P | R | F1 | P | R | F1 | P | R | F1 |
| Ours | 0.923 | 0.068 | 0.127 | 0.941 | 0.101 | 0.182 | **0.947** | 0.165 | **0.281** |
| BootEA | 0.330 | 0.077 | 0.125 | 0.321 | 0.124 | 0.179 | 0.309 | **0.186** | 0.232 |

– Our method always achieves high precision but low recall, which is sometimes even lower than that of BootEA. This is because we only used approximate precision as the likelihood in our EM strategy, which means that our method can mine high-quality matched entities but could sacrifice recall. Besides, our method requests a large number of high-quality labeled data to generate reasonable rules, so recall may not be guaranteed in small datasets, but it could be solved in large-scale datasets.
– When increasing the number of labeled data in training, the F1-score and recall of our method become higher, but precision may not. More labeled data can provide more rules to find new matched entities, so recall can become higher, but may also introduce noisy rules even labeled data are correct, which lowers the precision. Overall, more labeled data can bring larger positive effects than negative ones, which is reflected by F1-score.
– BootEA does not have a good performance, especially the precision is quite low. The main reason is that the triples corresponding to each property (i.e., relation) are very sparse especially we did not have aligned properties, which affects the quality of learned embeddings.

We also tested the link discovery method WOMBAT [27] which is a semi-supervised learning method for entity matching only using positive training data, but the result is quite poor (F1 is below 0.01). The main reason we hypothesize that WOMBAT is well tuned on some small domain-specific datasets

**Table 9** Overall statistics on extraction results

| **Items** | **Baidu Baike** | | **Hudong Baike** | | **Chinese Wikipedia** | |
|---|---|---|---|---|---|---|
| Entities | **14,307,056** | | 5,521,163 | | 903,462 | |
| have abstracts | 10,434,530 | 72.9% | 3,728,441 | 67.5% | 698,591 | **77.3**% |
| have categories | 11,458,878 | 80.1% | 3,256,294 | 59.0% | 732,273 | **81.1**% |
| have infoboxes | 11,811,513 | **82.6**% | 1,644,300 | 29.8% | 257,452 | 28.5% |
| | | per ent. | | per ent. | | per ent. |
| Categories | 32,276,892 | 2.25 | 16,363,407 | 2.96 | 3,147,262 | **3.48** |
| External Links | 10,780,841 | **0.75** | 2,013,874 | 0.36 | 634,299 | 0.70 |
| Images | 13,056,860 | 0.91 | 5,480,107 | **0.99** | 334,683 | 0.37 |
| Infobox Properties | 48,838,895 | 3.41 | 14,134,668 | 2.56 | 9,588,551 | **10.61** |
| Internal Links | 47,190,296 | 3.30 | 95,684,341 | **17.33** | 12,756,883 | 14.12 |

in Ontology Alignment Evaluation Initiative[16], but when faced with relatively large-scale multi-domain datasets, string similarities used in WOMBAT are not enough to handle such complex heterogeneities.

## 6.2 Evaluation of Knowledge

After applying our proposed framework on the whole Baidu Baike, Hudong Baike and Chinese Wikipedia, the obtained knowledge graph has 2,466,265,387 RDF triples and 17,283,702 distinct entities (equivalent entities only count once). We publish this knowledge graph[17] on the Web as a new version of the Zhishi.me project, called Zhishi.me2. The details of the generated knowledge are introduced in this subsection.

### *6.2.1 Results of Knowledge Extraction*

For regular extraction, encyclopedia articles from Baidu Baike and Hudong Baike were crawled in August, 2018. The dump version of Chinese Wikipedia is 20180828. Live extraction is executed everyday once after October 1, 2018. The statistics here were collected at November 1, 2018. Table 9 shows the statistics of extracted content in detail.

Baidu Baike has the largest number of entities and the most entities containing infoboxes. It indicates that this data source has a wide coverage of Chinese entities. Compared with Baike Baike and Hudong Baike, the proportion of the entities containing abstracts and that of the entities having categories are the highest. Note that each decimal following the absolute number in Table 9 is the fractions that divided by the total entity number in each data source. Considering the average extracted contents owned per entity, each online encyclopedia has its own advantage. Baidu Baike has the most external links (0.75) per entity. Hudong Baike has relative more images (0.99) and internal links (17.33) per entity than others, while Chinese Wikipedia contains more categories (3.48) and infobox properties (10.61) per entity.

[16] `http://oaei.ontologymatching.org/`
[17] `http://zhishi.me/`

**Table 10** Frequent used infobox properties of each data source

| Baidu Baike | | Hudong Baike | | Chinese Wikipedia | |
|---|---|---|---|---|---|
| Chinese Name | 561,732 | Chinese Name | 252,234 | Full Name | 44,728 |
| Date of Birth | 374,496 | Sex | 125,424 | Population | 39,323 |
| Nationality | 352,739 | Date of Birth | 102,320 | Website | 35,943 |
| Birthplace | 283,390 | Nationality | 98,223 | Address | 30,392 |
| Foreign Name | 235,361 | Occupation | 90,325 | Language | 29,352 |
| Occupation | 213,940 | Home Town | 81,328 | Height | 27,438 |
| Nationality | 147,324 | Era | 80,234 | Kana | 26,342 |
| Category | 115,974 | English Name | 75,305 | Hiragana | 26,223 |
| Scientific Name | 103,472 | Kingdom | 76,156 | Director | 24,237 |
| Alias | 98,892 | Achievements | 70,251 | Romanization | 20,100 |
| Kingdom | 92,749 | Category | 65,732 | Prefectures | 17,045 |
| Family | 90,327 | Scientific Name | 61,359 | Achievements | 14,994 |
| Achievements | 88,791 | Class | 58,434 | Starring | 12,345 |
| Representative Works | 83,235 | Phylum | 56,332 | Scenarist | 9,949 |
| Full Name | 82,203 | Family | 49,838 | Area | 8,346 |

Since an infobox represents a summary of information about the given entity as property-value pairs, *Infobox Properties* is taken as the most valuable knowledge in the generated knowledge graph, and we listed the most frequently used infobox properties in Table 10. The original properties are written in Chinese but we translated them into English in Table 10. The types of entities that have infoboxes can be easily inferred from these frequently used properties. Hudong Baike has a large quantity of persons and organisms described in minute detail. Similarly, most listed Baidu Baike properties manifest different facets of somebodies or living things. Chinese Wikipedia also describes lots of people, but in a little different perspective. In addition, we can also find featured properties for geographical regions (e.g., population and area) and films (e.g., director and starring).

Another important extracted content is *Entity Types*, which indicates the classification information of entities. Top-5 types in each data source are listed in Table 11. Total number of entities that share these types accounts for over one third entities that have type information. It is easy to find that the top-5 types have many overlaps in the three online encyclopedias. This suggests that it is necessary to perform knowledge linking among these data sources.

**Table 11** Top-5 types of each data source (Expr.: Expressions)

| Baidu Baike | | Hudong Baike | | Chinese Wikipedia | |
|---|---|---|---|---|---|
| Persons | 1,064,648 | Persons | 302,435 | Persons | 71,347 |
| Works | 894,692 | Places | 288,432 | Organisms | 66,335 |
| Places | 826,086 | Works | 215,003 | Places | 50,432 |
| Organisms | 783,692 | Words and Expr. | 209,632 | Organizations | 43,239 |
| Words and Expr. | 464,407 | Organisms | 182,814 | Works | 21,281 |
| Subtotal | 4,033,525 | Subtotal | 1,198,316 | Subtotal | 252,634 |
| Account for | 35.2% | Account for | 36.8% | Account for | 34.5% |

### *6.2.2 Results of Knowledge Linking*

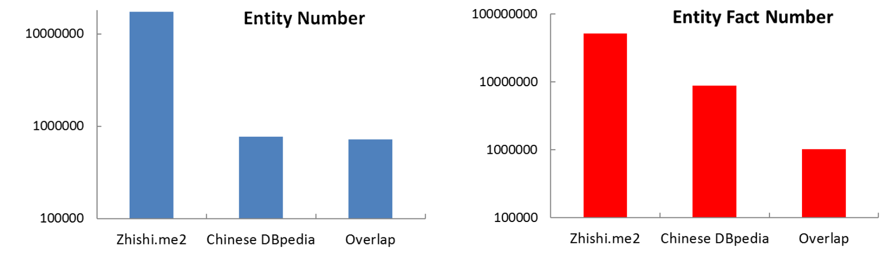
After applying our strategies of lightweight entity matching to the whole Baidu Baike, Hudong Baike and Chinese Wikipedia, we obtained 1,049,382 pairs of equivalent entities between Baidu Baike and Hudong Baike, 299,527 pairs of equivalent entities between Chinese Wikipedia and Baidu Baike, and 196,143 pairs of equivalent entities between Chinese Wikipedia and Hudong Baike.

Then, to discover more matched entities among these online encyclopedias, we ran our proposed method of semi-supervised entity matching on all entities, and the input is all pairs of known matched entities in which each entity should have at least one infobox property. We got 252,191 new pairs of matched entities and 8,579 pairs of matched properties between Baidu Baike and Hudong Baike, 117,947 new pairs of matched entities and 5,029 pairs of matched properties between Baidu Baike and Chinese Wikipedia, and 114,567 new pairs of matched entities and 2,641 pairs of matched properties between Hudong Baike and Chinese Wikipedia. Since there is no ground truth available, we evaluated all of them manually. For each pair of online encyclopedias, we randomly selected 200 pairs of matched entities and 200 pairs of matched properties, and invited five graduated students to participate in judgment. We provided them two choices namely *positive* and *negative* to label each pair of matched entities and that of matched properties. We computed 1) the Fleiss' Kappa [9] to evaluate the labeling consistence; 2) the average precision with the Wilson interval [4] at $\alpha = 5\%$ ($\alpha$ is the significance level) to generalize our findings on the small subsets to the whole set of matched entities and that of matched properties. The results show the high quality of matched entities and matched properties as follows:

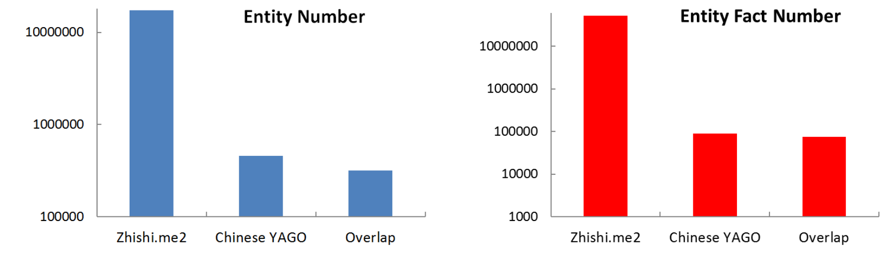
- For Baidu Baike and Hudong Baike, the average precisions of pairs of matched entities and pairs of matched properties are $95.32\% \pm 2.76\%$ (the Fleiss' Kappa: 0.833) and $94.25\% \pm 3.08\%$ (the Fleiss' Kappa: 0.730), respectively.
- For Baidu Baike and Chinese Wikipedia, the average precision for matched entities is $93.66\% \pm 3.24\%$ (the Fleiss' Kappa: 0.889) and that for matched properties is $92.87\% \pm 3.43\%$ (the Fleiss' Kappa: 0.816).
- For Hudong Baike and Chinese Wikipedia, we got the average precision of pairs of matched entities $96.6\% \pm 2.32\%$ (the Fleiss' Kappa: 0.857) and that of pairs of matched properties $93.95\% \pm 3.16\%$ (the Fleiss' Kappa: 0.814).

### *6.2.3 Comparison with Other Knowledge Graphs*

We compared Zhishi.me2 with Chinese DBpedia and Chinese YAGO in terms of entities and the entity facts derived from *Infobox Properties*. Here, equivalent entities and entity facts from different online encyclopedias only count once. Fig. 7 (a) shows the comparison results between Zhishi.me2 and Chinese DBpedia. As for the entity number, Zhishi.me2 has more than 17 million entities, but Chinese DBpedia only has around 0.77 million entities. Zhishi.me2 covers



(a) Comparison with Chinese DBpedia



(b) Comparison with Chinese YAGO

**Fig. 7** Comparison between Zhishi.me2 and other knowledge graphs

almost all entities in Chinese DBpedia, because we also extracted knowledge from Chinese Wikipedia. The minor difference exists in the version difference of the dumps of Chinese Wikipedia that the dump we extracted is 201808 and the latest DBpedia was published in 2016. Regarding the number of entity facts, Zhishi.me2 has more than 51 million entity facts, but Chinese DBpedia only has about 8.8 million entity facts. Zhishi.me2 only covers 1/8 entity facts in Chinese DBpedia. This is because only 1/10 entity facts in Chinese DBpedia utilize Chinese properties as the predicates, and others uses English properties. Although we leveraged cross-lingual links in Wikipedia to map English properties to Chinese ones, but the coverage is low.

Fig. 7 (b) shows the comparison results between Zhishi.me2 and Chinese YAGO. Note that YAGO does not have a Chinese version, but it contains Chinese labels of some English entities. We can see that Chinese YAGO only contains 0.45 million entities, around two-thirds of which are covered by Zhishi.me2. To acquire Chinese entity facts from YAGO, we first replaced the English entity labels in triples with Chinese labels. We then mapped the English relations in YAGO to the English properties in DBpedia by the mapping file[18] provided by the YAGO project, and also used the cross-lingual links in Wikipedia to get Chinese properties. As a result, Chinese YAGO only has 89 thousand entity facts, most of which are covered by Zhishi.me2.

The above comparison results fully reflect that Chinese knowledge is quite sparse in DBpedia and YAGO, and mining non-English knowledge from not just Wikipedia, but multiple online encyclopedias is valuable.

[18] `https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/linking/`

## 7 Conclusions

In this paper, we presented all technical details of each part in our proposed general framework for knowledge graph construction from multiple online encyclopedias, including regular extraction, live extraction, lightweight entity matching and semi-supervised entity matching. We applied our framework to three largest Chinese online encyclopedias: Baidu Baike, Hudong Baike and Chinese Wikipedia. The resulting knowledge graph Zhishi.me2 contains large-scale RDF triples and covers wide-range entities from various domains, which shows the effectiveness of our framework. Evaluation results also show the superiority of the proposed method of live extraction and semi-supervised entity matching when comparing with state-of-the-art baselines.

As for the future work, since current knowledge extraction focuses on mining RDF triples from semi-structured data, we plan to study slot filling techniques to discover more knowledge from unstructured text in the article pages of online encyclopedias. Besides, quality control is an important task of knowledge graph maintenance, so we will apply inconsistency detection and repair, as well as human-centered crowdsourcing techniques to knowledge graph construction, to guarantee the high quality of the continuously updated knowledge graph. Since the proposed framework is actually language-independent, we will also apply this framework to the online encyclopedias of other languages (e.g., Korean and Spanish) to contribute more non-English knowledge graphs.

## 8 Acknowledgements

This work was supported in part by National Key R&D Program of China (2018YFC0830200, 2017YFB1002801) and National Natural Science Foundation of China Key Project (U1736204).

## References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: Proc. of VLDB, vol. 1215, pp. 487–499 (1994)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data-The Story So Far. International Journal on Semantic Web and Information Systems **5**(3), 1–22 (2009)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia-A Crystallization Point for the Web of Data. Journal of Web Semantics **7**(3), 154–165 (2009)
4. Brown, L.D., Cai, T.T., DasGupta, A.: Interval Estimation for a Binomial Proportion. Statistical Science pp. 101–117 (2001)
5. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment. In: Proc. of IJCAI, pp. 1511–1517 (2017)
6. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological) pp. 1–38 (1977)
7. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2007)
8. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge, MA (1998)

9. Fleiss, J.L.: Measuring Nominal Scale Agreement among Many Raters. Psychological bulletin **76**(5), 378 (1971)
10. Hellmann, S., Stadler, C., Lehmann, J., Auer, S.: DBpedia Live Extraction. In: OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", pp. 1209–1223 (2009)
11. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation **9**(8), 1735–1780 (1997)
12. Hu, W., Chen, J., Qu, Y.: A Self-Training Approach for Resolving Object Coreference on the Semantic Web. In: Proc. of WWW, pp. 87–96 (2011)
13. Hu, W., Jia, C.: A bootstrapping approach to entity linkage on the Semantic Web. Journal of Web Semantics **34**, 1–12 (2015)
14. Jin, H., Li, C., Zhang, J., Hou, L., Li, J., Zhang, P.: XLORE2: Large-scale Cross-lingual Knowledge Graph Construction and Application. Data Intelligence **1**(1), 77–98 (2019)
15. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web **6**(2), 167–195 (2015)
16. Liang, J., Zhang, S., Xiao, Y.: How to Keep a Knowledge Base Synchronized with Its Encyclopedia Source. In: Proc. of IJCAI, pp. 3749–3755 (2017)
17. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: A Knowledge Base from Multilingual Wikipedias. In: Proc. of CIDR (2013)
18. Mikolov, T., Zweig, G.: Context Dependent Recurrent Neural Network Language Model. In: Proc. of SLT, pp. 234–239 (2013)
19. Navigli, R., Ponzetto, S.P.: BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence **193**, 217–250 (2012)
20. Nentwig, M., Hartung, M., Ngonga Ngomo, A.C., Rahm, E.: A survey of current link discovery frameworks. Semantic Web **8**(3), 419–436 (2017)
21. Ngomo, A.C.N., Auer, S.: LIMES-A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In: Proc. of IJCAI, pp. 2312–2317 (2011)
22. Nikolov, A., Uren, V., Motta, E.: KnoFuss: A Comprehensive Architecture for Knowledge Fusion. In: Proc. of K-CAP, pp. 185–186 (2007)
23. Niu, X., Rong, S., Wang, H., Yu, Y.: An Effective Rule Miner for Instance Matching in a Web of Data. In: Proc. of CIKM, pp. 1085–1094 (2012)
24. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi. me-Weaving Chinese Linking Open Data. In: Proc. of ISWC, Part II, pp. 205–220 (2011)
25. Rico, M., Mihindukulasooriya, N., Gómez-Pérez, A.: Data-Driven RDF Property Semantic-Equivalence Detection Using NLP Techniques. In: European Knowledge Acquisition Workshop, pp. 797–804 (2016)
26. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
27. Sherif, M.A., Ngomo, A.C.N., Lehmann, J.: WOMBAT–A Generalization Approach for Automatic Link Discovery. In: European Semantic Web Conference, pp. 103–119 (2017)
28. Sun, Z., Hu, W., Li, C.: Cross-Lingual Entity Alignment via Joint Attribute-Preserving Embedding. In: Proc. of ISWC, Part I, pp. 628–644 (2017)
29. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping Entity Alignment with Knowledge Graph Embedding. In: Proc. of IJCAI, pp. 4396–4402 (2018)
30. Völker, J., Niepert, M.: Statistical Schema Induction. In: Proc. of ESWC, pp. 124–138 (2011)
31. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and Maintaining Links on the Web of Data. In: Proc. of ISWC, pp. 650–665 (2009)
32. Wang, Z., Li, J., Wang, Z., Li, S., Li, M., Zhang, D., Shi, Y., Liu, Y., Zhang, P., Tang, J.: XLore: A Large-scale English-Chinese Bilingual Knowledge Graph. In: Proc. of ISWC (Posters & Demos), vol. 1035, pp. 121–124 (2013)
33. Werbos, P.J.: Backpropagation Through Time: What It Does and How to Do It. Proceedings of the IEEE **78**(10), 1550–1560 (1990)
34. Wu, F., Weld, D.S.: Autonomously Semantifying Wikipedia. In: Proc. of CIKM, pp. 41–50 (2007)
35. Wu, T., Qi, G., Li, C., Wang, M.: A Survey of Techniques for Constructing Chinese Knowledge Graphs and Their Applications. Sustainability **10**(9), 3245 (2018)

36. Wu, T., Qi, G., Luo, B., Zhang, L., Wang, H.: Language-Independent Type Inference of the Instances from Multilingual Wikipedia. International Journal on Semantic Web and Information Systems **15**(2), 22–46 (2019)
37. Xu, B., Xu, Y., Liang, J., Xie, C., Liang, B., Cui, W., Xiao, Y.: CN-DBpedia: A Never-Ending Chinese Knowledge Extraction System. In: Proc. of IEA/AIE, pp. 428–438 (2017)
38. Zhang, Z., Gentile, A.L., Blomqvist, E., Augenstein, I., Ciravegna, F.: Statistical Knowledge Patterns: Identifying Synonymous Relations in Large Linked Datasets. In: Proc. of ISWC, Part I, pp. 703–719 (2013)
39. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative Entity Alignment via Joint Knowledge Embeddings. In: Proc. of IJCAI, pp. 4258–4264 (2017)