Detecting Synonymous Predicates from Online Encyclopedia with Rich Features

Zhe Han, Yansong Feng, Dongyan Zhao

{hanzhe1992, fengyansong, zhaodongyan}@pku.edu.cn

Abstract The integration of Linked Open Data faces great challenges on the semantic level, despite unified data models. Inappropriate use of ontology concepts, namely predicates, impedes knowledge discovery. Although predicate unification is one of the most crucial steps when building structured knowledge base, little effort has been put forward. In this paper, we propose a supervised approach to detect synonymous predicates. Our detection focuses on feature selection and their effectiveness analysis. We not only leverage different resources such as Wikipedia, Freebase, but also use different word embeddings to represent predicates. The experimental results indicate that wikitext defined by Wikipedia and predicate surface form are most useful features.

1 Introduction

1.1 Motivation

Recently, much effort has been devoted to automatically building structured KBs from English version of Wikipedia, such as Freebase [4] , DBpedia [3] and YAGO [12]. These KBs, unlike Wikipedia itself, extract facts in structured form of subject-predicate-object (s-o-p) triples from infoboxes in Wikipedia. However, Wikipedia pages are maintained by volunteers around the world. People from different backgrounds may use different expressions to convery the same idea, causing different surface forms of predicates which are semantically identical. The incorrect use of predicates causes either low recall of extractor or redundant relations in structured KBs.

Table 1: predicates containing 邮政 (postcode) in Chinese Wikipedia infoboxes

predicate	INSEE/邮政编码	邮政编码	邮政区号	邮政编号	邮政简称	ISO 3166-2 邮政简写	邮政信箱	美国邮政编号	美国邮政编码
frequency	7257	5671	127	14	10	3	2	2	2
predicate	邮政编码首字母	邮政分区	邮政	邮政代码	邮政号码	邮政编码 FSA	邮政缩写	邮政编号字母	
frequency	1	1	1	1	1	1	1	1	

There are 14,000 different predicate surface forms in Chinese Wikipedia [¶]. Many predicates are in fact synonymous. For example, there are 17 predicates containing 邮政 (*postcode*) in Chinese Wikipedia infoboxes, shown in Table 1. Most of them represent 'postcode'. When an editor is submitting a new attribute of an entity, the system should provide the editor with candidate predicates. Besides, query expansion also requires synonymous predicates recommendation. The tremendous predicate list makes it impossible to get rid of duplicate predicates manually. It's urgent to put forward an auto-detect method to find synonymous predicates in online encyclopedias, which helps improve the quanlity of structured KB's extractions.

1.2 Challenges

Predicate detection in online encyclopedias differs from that on Linked Open Data(LOD), such as DBpedia and Freebase. Objects in online encyclopedias are often non-standard, making it difficult to use. Moreover, predicates are various due to different backgrounds of editors. Usage of global synonym databases is not sufficient as predicates are used in various KBs for various purpose by various editors. There are also different interpretations of predicates. Some predicates are too concrete while others are too general. Besides, many Chinese characters share a similar pronunciation, causing typos (mistaking characters with the same pronunciation) and transliteration differences (different characters chosen to represent the same pronunciation). As for Chinese predicates, there are fewer external resources like WordNet. The long tail [15] causes little information could be extracted from low frequent predicates.

1.3 Contribution

Earlier studies on structured KBs are not appropriate in the case for online encyclopedias. Our method is exactly designed for semi-structured online encyclopedias where objects are seldom linked entities. The contribution of this paper is fourfold. First, we leverage Wikipedia's wikitext for the first time to describe predicates. Besides, we extract many detailed information in Wikipedia and joint dumps and web pages together for the first time. Second, we propose various word-embeddings, varying from predicate types to predicate semantics. Third, we use linking information between Freebase schema and Wikipedia schema and use the better organized Freebase schema to describe predicates. Finally, we understand the predicate from these features.

The rest of this paper is organized as follows: In the next section we present related work with regard to synonymous predicates discovery. Next in Sect. 3 and Sect. 4, we introduce the resources and features used in our experiment. We evaluate the features from different perspectives in Sect. 5 and conclude in Sect. 6.

[¶] based on Chinese Wikipedia web pages in August 2014

2 Related work

Although it is a fundamental step in building structured KB, rare work has been done on this intractable problem. Many released KBs avoid predicate unification by using a predefined and limited predicate list, such as YAGO. There are less than 200 predicates in YAGO. You cannot find the screenwriter of any movies in YAGO because this relation has not be defined yet. Freebase indeed detect synonyms based on user domain expertise and co-occurrence of objects and subjects [13]. However, this method calls for user logs and well-structured KB, which can not be utilized by other KBs.

Most techniques for synonym detection derive from schema matching as data mining in the Semantic Web, associated with query expansion and synonym discovery. Others are based on different language processing and information retrieval techniques.

Mature methods in Semantic Web, such as frequent subgraph or subtrees analysis [9], are not suitable because no two different nodes in an RDF graph have the same URI. Instead, we consider the corresponding type of each URI as different URI may belong to the same type. Cafarella [5] presents a approach to detect synonyms among table attributes. However, the authors restrict attributes and ignore instance-based method because they concentrate only on extracted table schemata. So far, Abedjan [1] treats synonymous predicates detection as a association rule mining problem. Note that he works on structured DBpedia using linking information of objects and does not understand the predicates. This method is not appropriate for encyclopedias.

Baroni [2] and Wei [14] propose a common approach using co-occurrence of synonym candidates in web documents, based on the idea of synonymous word co-occur [8]. Naumann [11] proves the effectiveness of aggregate features and Li's work [10] shows that the performance using dictionaries only in real data is poor. In this case, we use multi features to capture predicate semantics.

Since only subject is normalized in encyclopedias, we use subject schema and NLP tools to discover synonyms, leveraging both the benefit of schema matching and semantic understanding.

3 Resources

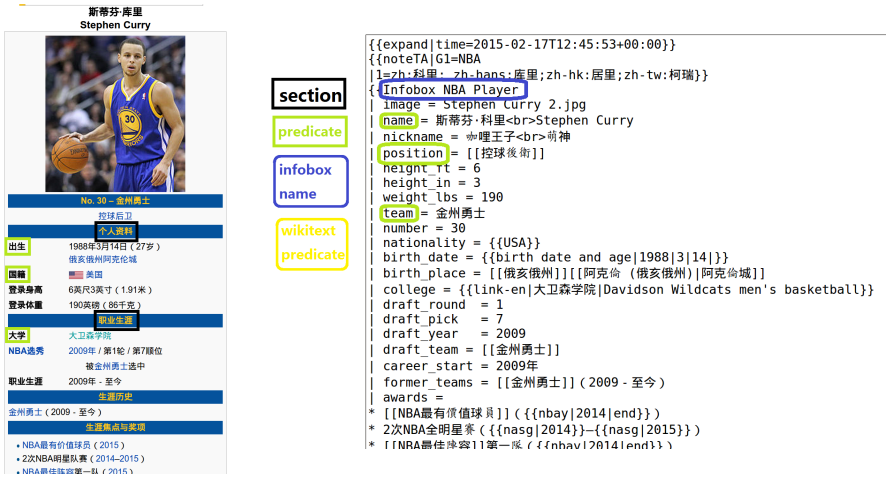

Figure 1: The web infobox (left) and wikitext infobox (right) of Stephen Curry in Chinese Wikipedia. wikiSection and wikiInfobox refer to 'section' and 'infobox name'. 'wikitext predicate' will be aligned to 'predicate' based on their attribute values. e.g. wikitext predicate 'nationality' is aligned to web predicate '国籍' because 'USA' and '美国' are the same.

Various resources have been used as features to present predicates, from inside and outside the KB. We leverage both web pages and dumps of Wikipedia in the experiment. Besides, bilingual dictionary and Freebase schema are used to represent each predicate. Our main dataset is a semi-structured KB ¶ (only subject is defined as an entity) with 3.5 millions s-p-o triples extracted from 33.8 thousands of infoboxes in Chinese Wikipedia [7], which contains 14,000 different predicates. The KB is open-domain and predicates in the KB that have same surface forms are considered the same in our experiment.

3.1 Wikipedia

In Wikipedia, there are mainly three parts of data to help evaluate the similarity between predicates, including *section names* in Wikipedia web pages and *wikitext-predicates*, *infobox names* in Wikipedia dumps [*]. Fig. 1 shows all the information in Wikipedia.

wikiText Wikitext, also known as wikicode, is a lightweight markup language used to write pages in Wikipedia. Infobox [16] is a template used to collect and present a

[*] available at *https://dumps.wikimedia.org/zhwiki/*

subset of information about its subject. All the wikitexts and infoboxes [17] mentioned in this paper are referred to Wikipedia's wikitexts and infoboxes.

wikiSection Attributes are often divided into different sections in Wikipedia infoboxes. As shown in Fig. 1, 个人资料 (*personal information*) is the section name of predicate 出生 (*birth*) and 登陆身高 (*listed hight*) while predicate 大学 (*college*) and *NBA* 选秀 (*NBA draft*) are in section 职业生涯 (*career information*).

wikiInfobox Actually, each infobox of an entity has a name, which can be extracted from wikitext. For example, predicate 国籍 (*nationality*) usually appears in infoboxes concerning people while predicate 编剧 (*writer*) appears in infoboxes concerning drama.

3.2 Freebase

Table 2: Categories of actor 张家辉 *in Freebase (mid=m.03cp9fl))*

people.person	award.award_winner	film.actor	film.editor
tv.tv_actor	award.award_nominee	common.topic	

Different predicates are usually associated with different kinds of entities [6]. Predicate categories can be represented by their corresponding subject categories. On the one hand, Wikipedia's build-in categories are too detailed to use. There are more than 190,000 categories in Chinese Wikipedia. In addition, there exist many confusing but frequent categories, such as 优良条目 (*good articles*) and 含有希伯来语的条目 (*articles containing Hebrew-language text*). On the other hand, Freebase provides find-grained category information for most entities and fortunately many Freebase entities have been linked to Wikipedia entities [‖]. For example, the category information of Hongkong film actor 张家辉 [† ‡] (*Nick Cheung*) in Freebase is shown in Table 2. We collect all the categories of Freebase entities that correspond to Chinese Wikipedia entities [§].

4 Features

Features are of great importance in our experiment. We not only use the surface form of predicate, but also extract many latent features inside Wikipedia and Freebase. Table 3 presents all the features used in the experiment.

[‖] The linking property in Freebase rdf dump is *Wikipedia.zh-cn_id* while the Freebase category predicate is *rdf:type*

[†] https://zh.wikipedia.org/wiki?curid=472824

[‡] http://www.freebase.com/m/03cp9fl

[§] the version of Freebase used in the experiment is 2013-06-02 (1.37 billion triples). We collected categories of 337042 entities in Freebase

Table 3: All features

surfaceForm	1.unigram$_{(0,1)}$	3.edit_distance$_{(0,1)}$	5.length_ratio
	2.unigram$_{(1,0)}$	4.edit distance$_{(1,0)}$	
Pinyin	6.pinyin_unigram$_{(0,1)}$	8.pinyin_edit_distance$_{(0,1)}$	10.pinyin_length_ratio
	7.pinyin_unigram$_{(1,0)}$	9.pinyin_edit_distance$_{(1,0)}$	
wikiText	11.wikiText-embedding	12 wikiText$_{(0,1)}$	13.wikiText$_{(1,0)}$
wikiSection	14.wikiSection-embedding	15.wikiSection$_{(0,1)}$	16.wikiSection$_{(1,0)}$
wikiInfobox	17.wikiInfobox-embedding	18.wikiInfobox$_{(0,1)}$	19.wikiInfobox$_{(1,0)}$
Freebase Category	20.Freebase_SVD-embedding	21. Freebase-embedding	

4.1 surfaceForm

The most straightforward features would be those extracted from surface forms of predicates. This kind of features express the character level similarity between two predicates. We first consider unigram overlap and explore two metrics, unigram$_{(0,1)}$ (feature 1) and unigram$_{(1,0)}$ (feature 2). Unigram$_{(1,0)}$ and unigram$_{(0,1)}$ scores between a predicate pair $(pred_1, pred_2)$ are defined as follows, while other features containing subscript $_{(1,0)}$ or $_{(0,1)}$ are defined in the same way as Eq. (1) and Eq. (2):

$$unigram_{(1,0)}(pred_1, pred_2) = \frac{character\_overlap(pred_1, pred_2)}{character\_count(pred_1)} \tag{1}$$

$$unigram_{(0,1)}(pred_1, pred_2) = \frac{character\_overlap(pred_1, pred_2)}{character\_count(pred_2)} \tag{2}$$

$$edit\_distance_{(0,1)}(pred_1, pred_2) = \frac{edit\_distance(pred_1, pred_2)}{character\_count(pred_1)} \tag{3}$$

We also compute edit distances of each pair of predicates (feature 3 and 4) in Eq. (3). Synonymous predicates usually have similar length in characters, which is taken into account by *length(shorter predicate)/length(longer predicate)* as character length ratio (feature 5).

4.2 Pinyin

Pinyin is the official phonetic system (and ISO standard) for transcribing Mandarin pronunciations into the Latin alphabets. There are many words in Chinese with different writing forms, conveying the same meaning. For example, 坐标 (*coordinate*) and 座标 (*coordinate*) are different predicate forms but actually the same. We use the most frequent pinyin string of each Chinese character to construct the pinyin representation for a predicate. Features in *Pinyin* (feature 6-10) are similar to features in *surfaceForm*. Compared to features in *surfaceForm*, characters are replaced with their corresponding pinyin strings while calculating the similarity scores.

4.3 wikiText

Table 4: The wikitext-predicate distribution of predicate 面积[**]

wikitext	面积	area	areatotal	*arearank*	*population total*	tarea	面积排名	area imperial	总面积	...
frequency	2860	1251	272	163	124	93	72	24	19	...

Wikipedia uses a large amount of rules to translate particular wikitext templates to the infoboxes we see on web pages. In our case, predicates in wikitext are aligned to predicates in web pages. The alignment is based on manual rules calculating the similarities between objects in web page triples and the attribute values of dumps' wikitexts. Accordingly, given a predicate, we can collect a set of aligned wikitext-predicates, with their alignment frequencies to this predicate. The alignment is not a one-to-one mapping, causing noise in the alignment. For example, the wikitext-predicates and their frequency aligned to predicate 面积 (*area*) are shown in Table 4. We defined it the *wikitext-predicate distribution* of predicate 面积.

Let $freq(p_i, wp_j)$ be the frequency of predicate p_i aligning to wikitext-predicate wp_j. Let $WL(p_i)$ be the wikitext-predicate set that has aligned to p_i. The wikiText$_{(0,1)}$ (feature 12) and wikiText$_{(1,0)}$ (feature 13) further characterize the overlap between the two predicates in an asymmetric way, defined in Eq. (4) and Eq. (5) :

$$wikiText_{(0,1)}(p_1, p_2) = \frac{\sum_{wp_j \in (WL(p_1) \cap WL(p_2))} freq(p_1, wp_j)}{\sum_{wp_j \in WL(p_1)} freq(p_1, wp_j)} \tag{4}$$

$$wikiText_{(1,0)}(p_1, p_2) = \frac{\sum_{wp_j \in (WL(p_1) \cap WL(p_2))} freq(p_2, wp_j)}{\sum_{wp_j \in WL(p_2)} freq(p_2, wp_j)} \tag{5}$$

The wikitext-embedding of each predicate p_i is a unit, sparse vector $v_i = (v_1^i, v_2^i, ..., v_M^i)$. M is equal to the number of different wikitext-predicates. v_j^i is the normalized frequency between predicate p_i and wikitext-predicate wp_j, representing their co-occurence, defined in Eq. (6). Feature 11 of each predicate pair is the cosine similarity of the two wikitext-embedding vectors.

$$v_j^i = freq(p_i, wp_j) / \sqrt{\sum_j freq^2(p_i, wp_j)} \tag{6}$$

4.4 wikiSection and wikiInfobox

Table 5: The wikiSection distribution of predicate 国家 (country)

wikiSection	概览	地理位置	基本资料	废除前地理位置	概况	赛事信息	概要	基层政权	位置	...
frequency	42483	2107	721	582	180	113	113	93	86	...

[**] alignment errors are in *red*

Table 6: The wikiInfobox distribution of predicate 国家 (country)

wikiInfobox	infobox_settlement	infobox_city	东亚男性历史人物	infobox_kommune	geobox	infobox_uk_place	...
frequency	30978	1997	430	293	204	202	...

The predicate synonyms should have similar sections and infobox names. We collect all the wikiSections and wikiInfoboxes of predicates and convert them to distribution vectors. For example, The wikiSection and wikiInfobox distribution of predicate 国家 (*country*) is shown in Table 5 and Table 6. wikiSection and wikiInfobox features are calculated in a similar way as *wikiText* features.

4.5 Bilingual Dictionary

Some synonymous predicates are in different surface forms mainly because of translation differences. Thus we translate the original Chinese predicates to their corresponding English expressions. This kind of features works well when one or both predicates is low frequent and less information could be extracted by other kind of features.

4.6 Freebase Category

For each predicate, we average the category vectors of entities that appear as subject of this predicate to generate category vectors of predicate. Since Freebase has a large mount of different categories, the raw category information will be very sparse. Therefore, we use two kinds of Freebase category embeddings. One uses the original category distribution vector while the other uses singular value decomposition (SVD) to transform each entity's category information to a 100-dimension unit vector.

Let $S_i = \{e_1^i, e_2^i, ..., e_{N_i}^i\}$ be the set of entities that has predicate p_i, $T(e_j^i)$ be the set of types of entity e_j^i in Freebase. The original category embedding of p_i is $F_i = (f_1^i, f_1^i, ..., f_N^i)$. N_i is size of S_i while N is the total number of categories in Freebase. f_i^j is the normalized frequency between predicate p_i and Freebase category $cate_j$, defined in Eq. (7). The Freebase-embedding (Feature 21) of predicate pair (p_i, p_j) is $F_i * F_j$. So does feature 20.

$$f_j^i = (\sum_{e_k \in S_i} \sum_{c_j \in T(e_k)} 1)/(\sum_j (f_j^i * f_j^i) \tag{7}$$

5 Experiment

We treat this task as a binary classification problem, that is, given a pair of predicates $pred_1$, $pred_2$, predicting whether these two predicates are synonyms.

The dataset is validated by three experts in computer science major. The first expert randomly selects predicate pairs and tag *0* or *1* to represent whether they are synonyms. Since the class distribution is highly skewed with most predicate pairs being negative, we select a balanced set of 1500 pairs with 1000 positive and 500 negative to avoid failures in training. Then the second expert tags on this balanced pairs and the last person only tags the inconsistent pairs. The result training set contains 1000 pairs (464 pairs are tagged *1*) of predicates while the test set contains 500 pairs (240 pairs are tagged *1*).

To evaluate features' validity, we present three experiments: In Sect. 5.1 we evaluate the classification performance using only one kind of features. In Sect. 5.2 we evaluate the classification performance using all except one kind of features at one time. In Sect. 5.3 we evaluate all the feature combinations and seek the feature combination that outperforms others. We use LibSVM (with kernel type of LINEAR and RBF), decision tree (C4.5), voted perceptron and AdaBoost as classifiers in each experiment. Compared to Abedjan's work [1], we deal with different resources (linked open data and online encyclopedia) using different methods, thus, we use different evaluation methods.

5.1 Single Feature Experiment

Table 7: The single feature accuracy

feature(dimension)	Accuracy				
	AdaBoost	LibSVM_RBF	LibSVM_LINEAR	C4.5	VotedPerceptron
Pinyin (5)	**0.662**	**0.664**	**0.610**	**0.666**	0.618
surfaceForm (5)	0.634	0.584	0.586	0.634	**0.626**
Bilingual Dictionary (2)	0.594	0.598	0.598	0.596	0.586
Freebase Category (2)	0.568	0.580	0.562	0.568	0.582
wikiText (3)	0.562	0.572	0.586	0.562	0.562
wikiSection (3)	0.518	0.526	0.522	0.518	0.532
wikiInfobox (3)	0.518	0.526	0.522	0.518	0.532

First we explore the effectiveness of each kind of features. For each classifier, We report the accuracy using only one kind of features, shown in Table 7.

wikiSection and *wikiInfobox* features are indistinctive because much of low frequent predicates do not have enough wikiSections and wikiInfoboxes to represent predicates properly. *Pinyin* feature is of great importance as expected. It takes spell mistakes and different forms of expression into consideration. *SurfaceForm* and *bilingual dictionary* are also reported as good single features.

5.2 Minus One Feature Experiment

Table 8: The minus one feature accuracy

reduced feature	Accuracy				
	LibSVM_RBF	AdaBoost	LibSVM_LINEAR	C4.5	VotedPerceptron
-surfaceForm	**0.634**	**0.642**	**0.634**	**0.604**	**0.624**
-wikiText	0.656	0.666	0.648	0.644	0.666
-wikiInfobox	0.680	0.670	0.676	0.664	0.670
-wikiSection	0.680	0.670	0.676	0.664	0.670
-Pinyin	0.688	0.666	0.688	0.668	0.676
-Freebase Category	0.684	0.686	0.696	0.672	0.668
-Bilingual Dictionary	0.698	0.666	0.678	0.682	0.692

In the second experiment we have evaluated the redundancy of each kind of predicate comparing other features. We first remove one kind of features and then evaluate the utility of remaining features. The detached kind of features is more likely to be redundant if the remaining features have higher accuracy. The results are shown in Table 8.

wikiText feature is only inferior to *surfaceForm* feature while *bilingual dictionary* performs poor. It shows the importance of wikiText. *wikiText* includes the bilingual information for its cross-linguistic property. It also indicates that the wikiText defined by *Wikipedia.org* is valid and irreplaceable in representing predicate. No matter what kind of classifier we use, *surfaceForm* and *wikiText* appear the top 2 features in this experiment. What's more, *bilingual dictionary* is usually the most useless kind of features. It is because *bilingual dictionary* and *Pinyin* features can be seen as a coarse combination of *surfaceForm* and *wikiText* features. Comparing to the first experiment, *wikiInfobox* and *wikiSection* take effect in complex feature combinations.

5.3 Best Feature Combination

Table 9: The top feature combinations by accuracy (RBF)

features	**accuracy**
pinyin, surfaceForm, wikiText, wikiSection, wikiInfobox, Freebase_category	0.698
pinyin, surfaceForm, wikiText, wikiInfobox, Freebase_category	0.694
pinyin, surfaceForm, wikiText, wikiSection, Freebase_category	0.694
surfaceForm, wikiText, wikiInfobox, Freebase_category	0.688
surfaceForm, wikiText, wikiSection, Freebase_category	0.688
surfaceForm, wikiText, wikiSection, wikiInfobox, bilingual_dict, Freebase_category	0.688

In our last experiment, we want to find the best feature combination. We use LibSVM with RBF kernel as classifier example. The other classifiers output similar results. The result is shown in Table 9. Our best accuracy is achieved with features: [*pinyin, surfaceForm, wikiText, wikiSection, wikiInfobox, Freebase_category*]. It corresponds to the previous two experiments: *surfaceForm* and *wikiText* are fundamentally useful while *wikiInfobox* and *wikiSection* show their efficacy in complex feature combinations.

6 Conclusion and Future Work

In this paper, we propose a full-fledged method on detecting predicate synonyms, including features extraction and comparison. It is groundwork for building Chinese structured KB. We exploit a mount of features, including linking information between Freebase and Wikipedia. Thorough study has been done on wikitext. Our experiment shows that the wikitext provides unique information comparing to normal features. Subject category information and section information are also essential features, which can be used by other online encyclopedias.

In online encyclopedias, only few predicates will be inserted or changed by editors to entities pages during a short time. Synonymous predicates can be calculated offline and we can only calculate the similarity between recently modified predicates and other predicates, which reduces computation resources. Another way to speedup our system is using part of distribution data. We find that the top three wikitext-predicates in Sect. 4.3 already account for most correct alignment. Hence, the time complexity of feature calculating can be approximately linear.

Objects, or attribute values in the KB have not be leveraged, except for wikitext-predicate alignment. They depict the predicates directly and may contribute much in predicting the predicate synonyms. Future work will explore the use of objects in predicate comparison. Predicate unification between different Chinese encyclopedias, such as baidu baike and Chinese Wikipedia will also be conducted.

References

[1] Abedjan, Z., Naumann, F.: Synonym analysis for predicate expansion. In: The Semantic Web: Semantics and Big Data, pp. 140–154. Springer (2013)

[2] Baroni, M., Bisi, S.: Using cooccurrence statistics and the web to discover synonyms in a technical language. In: LREC (2004)

[3] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia-a crystallization point for the web of data. Web Semantics: science, services and agents on the world wide web 7(3), 154–165 (2009)

[4] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247–1250. ACM (2008)

[5] Cafarella, M.J., Halevy, A., Wang, D.Z., Wu, E., Zhang, Y.: Webtables: exploring the power of tables on the web. Proceedings of the VLDB Endowment 1(1), 538–549 (2008)

[6] Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: EMNLP-CoNLL. vol. 7, pp. 708–716 (2007)

[7] Denoyer, L., Gallinari, P.: The wikipedia xml corpus. In: Comparative Evaluation of XML Information Retrieval Systems, pp. 12–19. Springer (2007)

[8] Harris, Z.S.: Distributional structure. Word (1954)

[9] Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on. pp. 313–320. IEEE (2001)

[10] Li, W.S., Clifton, C.: Semint: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. Data & Knowledge Engineering 33(1), 49–84 (2000)

[11] Naumann, F., Ho, C.T., Tian, X., Haas, L.M., Megiddo, N.: Attribute classification using feature analysis. In: icde. vol. 271 (2002)

[12] Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web. pp. 697–706. ACM (2007)

[13] Tan, C.H., Agichtein, E., Ipeirotis, P., Gabrilovich, E.: Trust, but verify: predicting contribution quality for knowledge base construction and curation. In: Proceedings of the 7th ACM international conference on Web search and data mining. pp. 553–562. ACM (2014)

[14] Wei, X., Peng, F., Tseng, H., Lu, Y., Dumoulin, B.: Context sensitive synonym discovery for web search queries. In: Proceedings of the 18th ACM conference on Information and knowledge management. pp. 1585–1588. ACM (2009)

[15] Wu, F., Hoffmann, R., Weld, D.S.: Information extraction from wikipedia: Moving down the long tail. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 731–739. ACM (2008)

[16] Wu, F., Weld, D.S.: Autonomously semantifying wikipedia. In: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. pp. 41–50. ACM (2007)

[17] Wu, F., Weld, D.S.: Automatically refining the wikipedia infobox ontology. In: Proceedings of the 17th international conference on World Wide Web. pp. 635–644. ACM (2008)