PAPER

Entity Ranking for Queries with Modifiers Based on Knowledge Bases and Web Search Results

Wiradee IMRATTANATRAI[†a)], ***Nonmember***, **Makoto P. KATO**[†], ***Member***, **Katsumi TANAKA**[†], ***Nonmember***, ***and*** **Masatoshi YOSHIKAWA**[†], ***Member***

SUMMARY This paper proposes methods of finding a ranked list of entities for a given query (*e.g.* "Kennin-ji", "Tenryu-ji", or "Kinkaku-ji" for the query "ancient zen buddhist temples in kyoto") by leveraging different types of modifiers in the query through identifying corresponding properties (*e.g.* **established date** and **location** for the modifiers "ancient" and "kyoto", respectively). While most major search engines provide the entity search functionality that returns a list of entities based on users' queries, entities are neither presented for a wide variety of search queries, nor in the order that users expect. To enhance the effectiveness of entity search, we propose two entity ranking methods. Our first proposed method is a Web-based entity ranking that directly finds relevant entities from Web search results returned in response to the query as a whole, and propagates the estimated relevance to the other entities. The second proposed method is a property-based entity ranking that ranks entities based on properties corresponding to modifiers in the query. To this end, we propose a novel property identification method that identifies a set of relevant properties based on a Support Vector Machine (SVM) using our seven criteria that are effective for different types of modifiers. The experimental results showed that our proposed property identification method could predict more relevant properties than using each of the criteria separately. Moreover, we achieved the best performance for returning a ranked list of relevant entities when using the combination of the Web-based and property-based entity ranking methods.

key words: *entity ranking, property identification, knowledge base, web search*

1. Introduction

Entity search has been introduced as one of the recent emerging trends in most major search engines such as Google, Yahoo!, and Bing. This functionality supports search users in exploring entities more directly than finding entities from a collection of Web documents. The entity search is increasingly important as previous studies reported that about 71% of queries contain entity names, and at least 20-30% of them are simply entity names [1], [2]. The large amount of these kind of queries suggests that users often look for entity-related information while they are searching.

A major challenge in the entity search is to deal with complex queries since queries have been becoming longer for expressing much more specific information needs [3], [4]. Some long queries form sentences and contain types of entities to be retrieved as well as *modifiers*. For example, "*highest profitable* companies in japan" and "*upcoming american mystery* movies". Modifiers in these examples are "highest profitable", "upcoming", "american" and "mystery". Modifiers are usually used to further narrow down entities to be retrieved. However, the current entity search functionality does not filter or sort entities by considering many of the modifiers, especially modifiers that correspond to properties whose values are numerical, *i.e.* quantitative properties (*e.g.* the modifier "highest profitable" corresponding to the property **operatingIncome** of company entities), as it requires additional interpretation as well as more in-depth analysis in order to find relevant entities based on these modifiers. Moreover, the existing methods only focus on modifiers corresponding to properties whose values are categorical, *i.e.* qualitative properties (*e.g.* the modifier "mystery" corresponding to the property **genre** of movie entities), even though modifiers corresponding to quantitative properties are as useful as other modifiers for narrowing down the list of entities. Hence, it is necessary to interpret both types of modifiers to fully satisfy user needs in the entity search.

To overcome these limitations, we propose two entity ranking methods that return a ranked list of entities for queries with modifiers. The first proposed entity ranking method, *Web-based entity ranking*, directly finds highly relevant entities from Web search results returned in response to the query as a whole, and propagates the estimated relevance to the other entities. The second proposed entity ranking method, *property-based entity ranking*, ranks entities based on relevant properties of each modifier in the query. For identifying relevant properties, we propose a method based on a machine learning approach using our proposed seven criteria.

In the experiments, we used 50 queries to evaluate the effectiveness of our proposed methods. The results showed that our property identification method could identify more relevant properties when all criteria were combined using the SVM, and achieved the best performance for returning a ranked list of entities when using the combination of the Web-based and property-based entity ranking methods.

The contributions of this paper are:

1. We introduced two entity ranking methods for queries including modifiers: the Web-based entity ranking and property-based entity ranking.
2. We proposed a property identification method to be used in the property-based entity ranking based on our

Manuscript received November 15, 2017.
Manuscript revised April 20, 2018.
Manuscript publicized June 18, 2018.

[†]The authors are with Graduate School of Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan.
a) E-mail: wiradee@dl.kuis.kyoto-u.ac.jp

DOI: 10.1587/transinf.2017EDP7372

proposed seven criteria that are suitable for different types of modifiers.

3. We conducted experiments to evaluate the performance of the proposed methods and demonstrated the effectiveness of our methods over the existing methods.

This work is the extension of our previous work [5]. In this paper, we proposed four new criteria for identifying relevant properties and combined them with the previously proposed three criteria by using the SVM. We also performed the additional experiments for both property identification and entity ranking task.

The rest of the paper is organized as follows. Section 2 surveys related works on finding relevant entities and properties. Section 3 introduces our proposed methods, and Sect. 4 presents experimental results. Finally, Sect. 5 concludes the paper by outlining future work.

2. Related Work

This section introduces related works on finding a ranked list of entities and identifying relevant properties.

One of the most related works is *keyword search* over structured and semi-structured data [6]–[9]. Elbassuoni *et al.* proposed a retrieval model for the keyword search over RDF graphs. This retrieval model retrieves a set of subgraphs in which their elements (or entities) match with the keyword queries, and ranks them by considering the predicates (or properties) based on statistical language-models [6].

In addition to the keyword search over RDF, systems such as DISCOVER [7], DBXplorer [8] and BANKS [9] were developed to support the keyword search over relational databases. The idea is to return tuples containing given keywords using joining operations between relations in databases. For DISCOVER [7] and DBXplorer [8], the returned tuples are ranked by the number of joins required, whereas BANKS [9] ranks them based on the edge weight computation and prestige based ranking (*e.g.* PageRank [10]).

Similar to the keyword search problem, *question answering* over RDF graphs is also considered as one of the related works as they find answers (or entities) based on questions (or queries) with the use of the relevant properties for modifiers in the question [11], [12]. Zou *et al.* proposed methods to find relevant predicates or predicate paths (or properties) using supporting entity pairs for the relation phrases (or modifiers) [11], while Unger *et al.* proposed methods that use the pattern library extracted by the BOA framework [13] for finding relevant properties [12].

Furthermore, some related works are concerned with *entity ranking*, which has been addressed in some tracks in INEX and TREC [14]–[17]. The INEX entity ranking track is comprised of two tasks: entity ranking and entity list completion tasks. The entity ranking task expects systems to return relevant Wikipedia articles (or entities) in response to a given query where there is assumption that all entities have the corresponding pages in Wikipedia. Kaptein *et al.* proposed methods to rank Wikipedia entities by estimating relevant Wikipedia categories for a given query, and rank entities based on the query likelihood model with estimated categories [18]. Demartini *et al.* expanded queries for the entity ranking task in many ways such as using hierarchical relationship in the ontology and synonyms, and extracting named entities from the queries [19]. Balog *et al.* proposed a probabilistic framework, which can model not only keyword queries, but also the other inputs such as categories in which relevant entities should belong to and examples of relevant entities [20].

The entity track in TREC 2010 introduced another task called *related entity finding*. This task is related to the entity ranking tracks in INEX, but the main input is an entity name or a homepage of an entity and output is a list of homepages of entities. Bron *et al.* proposed methods for this task which reduce problems in ranking by using four components including co-occurrence, type filtering, context modeling, and homepage finding [21].

There are some more works related to entity ranking task. Zaragoza *et al.* proposed two methods using the candidate entities found in Wikipedia by a statistical entity extractor. The first proposed method utilized the entity containment graph with the inverse entity frequency, while the second method utilized Web search results with query-to-entity correlation measures where a set of retrieved documents by the query, entity, and their conjunction were used [22]. Pehcevski *et al.* utilized categories and the link structure of Wikipedia, and also exploited the link co-occurrence [23].

More study related to the problem of finding properties corresponding to modifiers was authored by Zhang *et al.* [24]. In this work, relevant properties (of *facets*) were estimated for a given query in order to provide better snippets for structured documents. While a machine learning approach was employed for finding relevant properties, most of the features were textual similarity that often used for learning to rank for Web search (*e.g.* TF-IDF or BM25).

The problem of returning a ranked list of entities in this research is different comparing to other related research works. The differences can be listed as 1) The query types in which we are interested in are different and 2) our returned ranked lists of entities are also associated with the ranking orders of the modifiers that correspond to the quantitative properties. Since our focused queries contain the modifiers that further indicate more specific characteristics of entities, these modifiers are not usually included in the element names within structured and semi-structured data, or even the Wikipedia articles. As a result, we cannot directly retrieve entities based on the similarity between the query and those predefined names. In this case, we not only use textual similarity, but also utilize Web search results using the original search queries, and the relevant properties of the modifiers. Moreover, as we interested in both modifiers corresponding to the qualitative and quantitative properties, the entities need to be filtered based on the qualitative properties as well as ordered based on the quantitative

properties. According to our survey, other related research works focus on modifiers corresponding to qualitative properties, which are required in filtering entities. On the other hand, the modifiers corresponding to quantitative properties are used to order the entities. This type of modifiers requires additional effort since we need to determine the ranking order of the entities according to the modifiers (*i.e.* ascending and descending order) and then rank the entities in the right order.

Moreover, the existing approaches for finding relevant properties such as the textual similarity are not suitable for modifiers corresponding to quantitative properties, for which we propose methods based on the co-occurrence of terms on the Web and property value distributions of entities.

3. Methodology

In this section, we propose two entity ranking methods: Web-based entity ranking and property-based entity ranking methods. We also propose a property identification method to identify relevant properties for the property-based entity ranking method.

3.1 Problem Definition

Before going through each proposed method, we formally define our problem in this subsection. Our problem is to return a ranked list of entities for a given query $q \in Q$. We are specifically interested in queries Q that contain an entity type name and modifiers. A modifier is an attributive term that describes a typical feature of entities. Modifiers can be categorized into two types in our research: ones corresponding to qualitative properties (*e.g.* "comedy", "new york" and "outsourcing"), and ones corresponding to quantitative properties (*e.g.* "oldest", "large" and "highest profitable").

Definition 1 (Qualitative properties): properties in which their property values are *nominal* or *ordinal* that cannot be measured, *e.g.* properties **genre**, **location**, and **award**.

Definition 2 (Quantitative properties): properties in which their property values are *interval* or *ratio* that can be measured, *e.g.* properties **established**, **weight**, and **operatingIncome**.

To return a list of entities for a given query, we utilize RDF data from a knowledge base that represents the relationship across entities E, entity types T, and properties P. Entities of the entity type $t \in T$ are denoted by E_t, entity types to which entity $e \in E$ belongs are denoted by T_e, and a set of properties of entity $e \in E$ are denoted by P_e.

In the following subsections, we describe each of our proposed methods for returning a ranked list of entities.

3.2 Web-Based Entity Ranking

The first proposed method directly finds highly relevant entities for a given query based on a Web search engine, and propagates the estimated relevance to the other entities based on the entity similarity. Since we assume that a given query contains entity type names, entities should be relevant if the names of the entity types they belong to are similar to the query. Thus, we use the BM25 to measure their similarity. Additionally, relevant entities may frequently appear in Web search results returned in response to the query as some Web pages list up entities and describe them with the same words in the query. We further explore relevant entities by using Co-HITS [25] for propagating the estimated relevance to the other similar entities.

3.2.1 BM25 Score between Query and Entity Type

We measure the similarity between query $q \in Q$ and entity type $t \in T$ using the BM25, which is denoted by $\mathrm{sim}(q,t)$. We regard entity types whose $\mathrm{sim}(q,t)$ is θ% (0.8 in our experiments) of the maximum score or larger as a set of relevant entity types as $T_q = \{t \,|\, \mathrm{sim}(q,t) \geq \theta \max_{t' \in T}(\mathrm{sim}(q,t'))\}$. Thus, candidates of relevant entities E_q should belong to these entity types: *i.e.* $E_q = \bigcup_{t \in T_q} E_t$.

3.2.2 Co-HITS Algorithm Using BM25 Score and Document Frequency

The BM25-based approach can only measure the relevance of entities in terms of their entity types, but not take into account modifiers in the query. Since there are Web pages introducing some entities with description including the same words in the query, one can know a *subset* of relevant entities by issuing the given query into a Web search engine and finding entity names within the search results. For example, there can be some Web pages introducing the highest profitable companies and can be retrieved through a Web search engine with query "highest profitable companies". As we emphasized, however, entities found by this approach can be only a subset of relevant entities. Thus, we propose a method to propagate the BM25 score and frequency on the Web search results with the Co-HITS algorithm.

We first issue the original search query q to a Web search engine and collect top D search results R_q (D was set to 20 in our experiments). For each candidate entity $e \in E_q$, we count the number of search results that include the entity name in their content as the document frequency. The document frequency is defined as $\mathrm{df}(e, R_q) = |\{r \,|\, r \in R_q \wedge e \in E_r\}|$ where E_r is a set of entities whose name is included in the content of search result r.

Given the BM25 scores of entity types and document frequency of entities, we utilize the Co-HITS algorithm [25] to combine and propagate these two scores based on two assumptions: 1) entity types are likely to be relevant if they contain relevant entities, and 2) entities are likely to be relevant if they belong to relevant entity types. As entities that have a high document frequency are likely to be relevant to query q, we can infer that entity types including such entities are also relevant. In a similar way, we can further infer

that entities included in relevant entity types are also relevant. The Co-HITS algorithm can be used to make these inferences and enables us to find more relevant entities and entity types that would be missed when using methods described above. The Co-HITS algorithm can be described as following equations:

$$c(q,t) = (1-\lambda_T)\,\mathrm{sim}(q,t) + \lambda_T \sum_{e\in E_q} w_{et} c(q,e), \tag{1}$$

$$c(q,e) = (1-\lambda_E)\,\mathrm{df}(e,R_q) + \lambda_E \sum_{t\in T_q} w_{te} c(q,t), \tag{2}$$

where $c(q,t)$ and $c(q,e)$ are the Co-HITS scores of entity types and entities, w_{et} is the edge weight from entity e to entity type t, w_{te} is the edge weight from t to e, and λ_T and λ_E are the parameters that control the effect of the BM25 score and document frequency on the Co-HITS score (1.00 and 0.25 were turned out to be the optimal, respectively, in our preliminary experiments and were used in our experiments). The edge weights indicate is-a relationship between the entity types and entities and are defined as $w_{et} = 1/|T_e|$ and $w_{te} = 1/|E_t|$ if entity e belongs to entity type t; otherwise, 0.

Computing $c(q,e)$ for each entity, we can finally obtain a list of ranked entities E_q^* in descending order of $c(q,e)$. Although this approach is simple and effective as can be seen in our experiments, we can use a complementary method, the property-based entity ranking method, with this Web-based entity ranking method and further improve the retrieval performance.

3.3 Property-Based Entity Ranking

This proposed method involves with three consecutive tasks: 1) modifier detection (detecting terms in a given search query to be modifiers), 2) property identification (identifying a set of relevant properties for modifiers), and 3) entity ranking by relevant properties (ranking entities using a set of relevant properties).

3.3.1 Modifier Detection

We firstly need to detect the modifiers in the query. To this end, we assume that the modifiers are terms that are not used for indicating the relevant entity types, *i.e.* ones that do not contribute to the BM25 scores of the relevant entity types. Specifically, for each term w in query q, we sum up all the BM25 scores of the entity types in T_q that contain term w in their entity type names, as $s(w) = \sum_{t\in T_q \wedge w\in n(t)} \mathrm{sim}(q,t)$, where $n(t)$ represents a set of terms in the name of entity type t. Then, we select term w as a modifier only if the sum of the BM25 scores for term w is relatively low. The set of modifiers in query q is denoted by M_q and obtained as $M_q = \left\{ w \,\middle|\, s(w) \le \phi \sum_{w'\in q} s(w') \right\}$, where ϕ is a parameter that we set to 0.2 in our experiments. The left term in the inequality is the sum of BM25 scores for term w, while the right term is that for all the terms in the query.

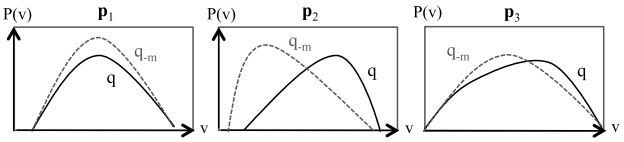


Fig. 1 Distributions of q and q_{-m} for each property (p_j) where x-axis (v) represents the property value of p_j, and y-axis ($P(v)$) represents probability density of property value v. Here, largest distribution difference can be seen with property p_2, which is possibly relevant to modifier m.

For each modifier $m \in M_q$, we attempt to find a corresponding property from properties $P_{E_q^*}$ ($= \bigcup_{e\in E_q^*} P_e$) that can appropriately describe the modifier.

3.3.2 Property Identification

The detected modifiers are used to identify relevant properties based on the seven criteria. The details of seven criteria are discussed in the following subsections.

Criterion 1: Frequency of Property Values The first criterion is to count the frequency of property values for each property p that contain modifier m. Some modifiers indicate a certain type of entities (*e.g.* "comedy" is a property value of relevant property **genre** and "new york" is included in the property values of the relevant property **location**) and often specifies the property values of relevant properties. The frequency of property values including modifier m is computed as follows:

$$\mathrm{freq}(q,m,p) = |\{e \,|\, e\in E_q^* \wedge m \in v(e,p)\}|, \tag{3}$$

where E_q^* is a set of candidate entities for query q and $v(e,p)$ represents a set of property values of property p for entity e.

Criterion 2: Co-occurrence of Modifiers and Properties on the Web The second criterion is to measure the co-occurrence of modifiers and property names on the Web. When people describe the characteristic of entities by modifiers (*e.g.* "old", "large", and "highest profitable"), they are likely to show their property values as evidence (*e.g.* "foundation date", "area size", and "income", respectively). Thus, a high co-occurrence between a modifier and a property name indicates that the modifier specifies a certain property value of the property. We compute how frequently modifier m co-occurs with property p by using Web search results from the original query q as follows:

$$\mathrm{co}(m,p) = \frac{|\,S_{mp}\,|}{|\,D_p\,|}, \tag{4}$$

where S_{mp} is the set of (adjacent) sentences in the Web pages containing modifier m and property p, and D_p is the set of web pages containing property p.

Criterion 3: Difference between Property Value Distributions The third criterion is to measure the difference in property value distributions of entities in search results

based on two queries 1) an original query (*e.g.* "large companion dog breeds") 2) a query without the modifier corresponding to qualitative properties (*e.g.* "large dog breeds" without the modifier "companion") or a query with the antonym of the modifier corresponding to quantitative properties (*e.g.* "small companion dog breeds" with the antonym of the modifier "large"). We expect that Web search engines will return search results containing different set of entities when using different queries. These different sets of entities would have different property values which relevant to terms in the query. Based on this assumption, we can measure the change in property value distributions of each property and estimate that properties that cause big changes are relevant to modifiers. Figure 1 shows some examples of pairs of property value distributions. We estimate that property p_2 is relevant to modifier m since the distributions for query q and q_{-m} are significantly different.

The property value distribution of property p for query q can be estimated as follows:

$$P_p^q(v) = \frac{|\{e \,|\, e \in E_{R_q} \wedge v \in v(e,p)\}|}{\sum_{v' \in V_p} |\{e \,|\, e \in E_{R_q} \wedge v' \in v(e,p)\}|}, \tag{5}$$

where v is a property value ($v \in V_p$; V_p is the set of all possible property values for property p), $v(e,p)$ is a set of property values of property p for entity e, and E_{R_q} is a set of entities included in the search results for query q (formally defined as $E_{R_q} = \bigcup_{r \in R_q} E_r$).

Letting q_{-m} be query q with the antonym of modifier m or without modifier m (*i.e.* $q - \{m\}$ if q is represented as a set of terms), the difference between property value distributions for queries q and q_{-m} is measured by Kullback-Leibler divergence as follows:

$$\begin{aligned} \mathrm{diff}(q,m,p) &= D(P_p^q \,||\, P_p^{q_{-m}}) \\ &= \sum_{v \in V_p} \log P_p^q(v) \frac{P_p^q(v)}{P_p^{q_{-m}}(v)}. \end{aligned} \tag{6}$$

Criterion 4: Similarity between Properties and Modifiers The fourth criterion is to compute the similarity between the name of each property p and modifier m. As some modifiers (*e.g.* the modifier "highest profitable") indicate the property names or the synonyms of the properties (*e.g.* the property **profit**, **income** and **revenue**), these properties have the same or similar meaning with the modifier and are likely to be relevant to the modifier m. To compute the similarity, we first constructed a distributional representation of words by word2vec [26] using the English version of Wikipedia articles (3,831,719 articles in total). Then, we compute the similarity between each property p and modifier m using the word vector representation by the word2vec model as follows:

$$\mathrm{sim}(m,p) = \frac{1}{|n(m)|} \sum_{w \in n(m)} \mathbf{v}_w^T \mathbf{v}_p, \tag{7}$$

where $n(m)$ represents a set of terms of modifier m, and $\mathbf{v}_w$ and $\mathbf{v}_p$ are word vectors of term w in $n(m)$ and property p, respectively.

Criterion 5: Co-occurrence of Major Property Value Types and Modifiers on the Web The fifth criterion is to measure the co-occurrence between major property value types and modifiers on the Web. Modifiers are not always described on the Web pages using the property names since the major property value types (*e.g.* **pounds**) can sometimes be adequate for interpreting the modifiers without explicitly indicating the property names (*e.g.* the property **weight**). For instance, within the top Web search results by the query "large dog breeds", the Web contents contain sentences describing the characteristic of large dogs by the modifier "large" and the major property value type **pounds** of the property **weight**, *i.e.* "**Large** *dogs typically tip the scales at 55 to 85* **pounds**" and "*Some groups define* **large** *dog breeds as those heavier than 100* **pounds**". The property is likely to be relevant to the modifier if the major property value type of this property frequently co-occurs with the modifier. The major property value type of each property (y_p^*) can be obtained as follows:

$$y_p^* = \underset{y \in Y_{V_p}}{\mathrm{argmax}} \, |\{e \,|\, e \in E_{R_q} \wedge y(e,p,v) = y\}|, \tag{8}$$

where Y_{V_p} is the set of all possible property value types for the property values of property p, E_{R_q} is a set of entities included in the search results for query q and $y(e,p,v)$ is the property value type of the property value v of the property p for the entity e.

After getting the major property value type of each property p, the co-occurrence of y_p^* with the modifier m can be computed as follows:

$$\mathrm{co}(m, y_p^*) = \frac{|S_{my_p^*}|}{|D_{y_p^*}|}, \tag{9}$$

where $S_{my_p^*}$ is a set of (adjacent) sentences in the Web pages containing modifier m and y_p^*, and $D_{y_p^*}$ is the set of web pages containing y_p^*.

Criterion 6: Difference between Co-occurrence of Property Values and Entity Names on the Web The sixth criterion is to measure the difference of the co-occurrence of property values and entity names on the Web for the original query q and the query without modifier q_{-m}. As the Web search results returned for each search query q contain the Web contents that are expected to be relevant to query q, the entities that are described in the content are likely to be followed by the property values of the properties which help supplementing the content to be more relevant to query q. For example, when issuing the query "old temples in japan", the temples could frequently appear in the contents of Web search results more with the established date than with the name of the founder. However, when issuing the query "temples in japan" without the modifier "old", we could see that the temples are less likely to appear with the established date in the Web content. Therefore, we compute the difference between co-occurrence of the name of entity

e with the property values of each property p on the Web by using the original query q and the query without modifier q_{-m}. We compute the co-occurrence of the name of entity e with the property values of each property p as follows:

$$C_p^q = \sum_{r \in R_q} \sum_{s_i \in r}^{|r|} |\{e \,|\, e \in E_{R_q} \wedge n(e) \subset s_i \wedge v(e,p) \subset s_i\}|, \quad (10)$$

where R_q is a set of Web search results by query q, s_i is a sentence in the content of the Web search result r, E_{R_q} is a set of entities included in the search results for query q, $n(e)$ is a set of terms in the name of entity e, and $v(e,p)$ is a set of property values of property p for entity e.

Then, we can compute the difference between the co-occurrence of the entity name with the property values of each property for the original query and the query without modifier on the Web as follows:

$$\mathrm{diff}_{\mathrm{co}}(q,m,p) = \frac{C_p^q}{C_p^{q-m}}, \quad (11)$$

If property p has high difference between the co-occurrence of the entity name with the property values of each property for the original query and the query without modifier on the Web, this property p is likely to be a relevant property for modifier m as the property values of property p are frequently used to explain the entities on the Web using the original query q.

Criterion 7: Frequency of Properties The seventh criterion is to compute the frequency of properties. This criterion is similar to the first criterion, but we count the number of entities whose property name appears as the modifier instead of the property value since it is possible that the modifier could directly indicate the property name as was mentioned earlier in the fourth criterion. We also used this seventh criterion because the fourth criterion measures only the similarity between the modifier and property names. In most cases, if the similarity between the modifier and property name is high, there is also high possibility that this property could be relevant to the modifier. However, we need to consider the number of entities that are associated with the property as well because if the number of entities is very low, we might not effectively rank the entities by using this property. The frequency of properties including modifier m is computed as follows:

$$\mathrm{freq}_{\mathrm{p}}(q,m,p) = |\{e \,|\, e \in E_q^* \wedge m \in p_e \wedge p \in p_e\}|, \quad (12)$$

where E_q^* is a set of candidate entities for a query q and p_e represents a set of properties associated with entity e.

Support Vector Machine using Seven Criteria for Identifying Relevant Properties Our proposed property identification method utilizes all seven criteria described previously as features in the SVM classification. Since each criterion is applicable for a different type of modifiers, all criteria need to be employed collectively for boosting performance in finding relevant properties for the modifier. We used a Radial Basis Function (RBF) as a kernel function to the SVM as it is commonly used to take into account the interaction of features in the SVM.

We selected the property with the highest predicted score by the SVM, $z(q,m,p)$, for each modifier $m \in M_q$ and finally obtained a set of properties as $P_q^* = \{\mathrm{argmax}_{p \in P_{E_q^*}} z(q,m,p) \,|\, m \in M_q\}$.

3.3.3 Entity Ranking by Relevant Properties

We rank entities using the relevant properties by firstly identifying the property type of the relevant property as different types of properties are used to differently rank the entities. Then, we assign the property value score for each property value of the property for entities depending on the ranking order correlated to each modifier. Lastly, we rank the entities by the sum of the property score from our proposed property identification method and property value scores of relevant properties.

Property Type Identification We identify the property type of each relevant property $p \in P_q^*$, *i.e.* whether it is a qualitative or quantitative property, since different property types need different ways of ranking the entities. For qualitative properties (*e.g.* the property **address**), we use their property values for filtering the entities, while the property values are used for ordering in the case of the quantitative properties (*e.g.* the property **founded**). To identify the property type, we use the property score of the three criteria used in Sect. 3.3.2 and also observe the major property value type of each property. We consider the property that has a higher property score of the first criterion than those of the other two criteria as a qualitative property. The first criterion can be applied to modifiers that appear as a part of property values. However, for the quantitative properties which their property values are usually numerical, it is unlikely that the modifiers (*e.g.* the modifier "ancient") appear as a part of property values. As a result, a higher score of the first criterion suggests that the property values of the properties are not numerical and that the properties are qualitative. Otherwise, the property is considered as quantitative as long as its major property value type is considered as numerical (*e.g.* date, centimeters and kilograms).

Property Value Scoring We assign the score to each property value of the relevant property for entities depending on the property type. In the case of qualitative properties, property value score can be simply assigned by the presence of the particular modifier m in the set of property values $v(e,p)$ of the property p for the entity e as defined by $s(e,m,p)$ where $s(e,m,p) = 1$ if $m \in v(e,p)$, otherwise, 0. On the other hand, we assign property value score for quantitative properties in the dissimilar way to the qualitative property. This is because the modifiers of relevant quantitative properties are associated with either ascending or descending order. We have to assign the property value based on the ranking order of the modifiers as we cannot

Table 1 Property value scoring for the quantitative properties depending on the ranking order of the modifier.

Ranking order	**Property value score** $s(e, m, p)$
Descending order	$\dfrac{v - \min_{e \in E_q^*} v(e, p)}{\max_{e \in E_q^*} v(e, p) - \min_{e \in E_q^*} v(e, p)}$
Ascending order	$\dfrac{\max_{e \in E_q^*} v(e, p) - v}{\max_{e \in E_q^*} v(e, p) - \min_{e \in E_q^*} v(e, p)}$

give a high property value score to the high property value of the quantitative property if the modifier is associated with ascending order (*e.g.* the property value 2017 of the relevant quantitative property **established** of the modifier "ancient"). By this reason, we need to decide the ranking order based on the modifier by calculating means of property values of the property p from two set of entities. As was explained earlier in Sect. 3.3.2, different set of entities will be returned by Web search engines when using different queries. Here, we use the same criteria to select two queries. Then, we compare the means by these two set of entities. If the mean by the original search query is larger, the modifier is associated with a descending order, otherwise an ascending order.

Afterwards, we assign the property value score to each property value v of the property p for a set of candidate entities E_q^* by the ranking order of the modifier using the equation in Table 1. By using these equations, the maximum property value will be assigned with the score as 1 for descending order, whereas the property value score for the maximum property value will be 0 for the ascending order.

Entity Ranking by Relevant Properties Finally, for each candidate entity $e \in E_q^*$, we rank them based on top R relevant properties for the modifier $m \in M_q$ in the query q by the sum of the property scores of the relevant properties P_q^* by the proposed property identification method in Sect. 3.3.2 and property value scores of relevant properties:

$$r(q, e) = \sum_{m \in M_q} \sum_{p \in P_q^*}^{R} z(q, m, p)\ s(e, m, p), \tag{13}$$

Then, we will get a list of ranked entities E_q^* based on relevant properties of modifiers indicated in the search query. Note that we can use both of the entity ranking methods described in Sects. 3.2 and 3.3 by summing their ranking score, *i.e.* $c(q, e) + r(q, e)$.

4. Experiments

In this section, we introduce evaluation methodology for our proposed methods, and show and discuss our experimental results.

In the experiments, we used RDF data from DBpedia as a knowledge base to evaluate our methods. We obtained a set of entity types, entities, properties and property values from DBpedia, which contains structured information derived from Wikipedia. We extracted pairs of an entity and an entity type by using the predicate <is a subject of> as the property, where the subjects and objects are treated as entity types and entities, respectively. For extracting properties and property values, we retrieved all triples whose the subjects are the entities, and treated the predicates as the property names and the objects as the property values. After the extraction, we got 974,417 entity types including 4,811,226 entities and 60,252 properties from all the entities in total.

We evaluated the resultant entity types obtained by Web-based entity ranking method for tuning parameters in the Co-HITS algorithm. We separately evaluated resultant properties and entity ranking for measuring the performance of our proposed methods in detail. We used 50 search queries that contain modifiers and an entity type name. Within these 50 queries, we detected 62 modifiers by the method explained in Sect. 3.3.1. 34 of them correspond to qualitative properties, while the remaining correspond to quantitative properties.

4.1 Evaluation of Co-HITS Parameters

To get the most optimal parameter setting for Co-HITS algorithm in the Web-based entity ranking, we evaluated the resultant entity types by the Web-based entity method. The relevance of retrieved entity types usually indicates that of entities in our assumptions. This preliminary experiment was performed with a subset of all the test queries (*i.e.* 40 queries out of 50 queries). For each query, we evaluated 10 entity types with the highest Co-HITS scores ($c(q, t)$) for 25 possible combinations of the two parameters involved in the Co-HITS algorithm, where λ_T and λ_E were set to 0.00, 0.25, 0.50, 0.75, and 1.00. Parameter λ_T controls the effect of the BM25 score of entity types, while parameter λ_E controls the effect of document frequency of entities on the Co-HITS score. As a result, we obtained 25 sets of ranked entity types for each query, and pooled the results for evaluation. The relevance assessment was carried out by three assessors. An entity type was considered relevant if at least one of its entities was relevant for a given query. The inter-rater agreement was measured by Fleiss' Kappa [27] and was considered to be moderate at 0.601. We computed the precision, recall, and F-measure based on the relevance given by three assessors.

Experimental Results Table 2 and 3 show the precision and recall of the resultant entity types for 25 combinations of parameter λ_E and λ_T, respectively. We noticed that the highest precision (0.659) was achieved when $\lambda_E = 0.00$ and $\lambda_T = 0.00$. Although the highest precision was achieved when $\lambda_T = 0.00$, the lowest recalls were obtained (0.198). Since $\lambda_T = 0.00$ indicates that the Co-HITS score was determined solely by the BM25 score, this result suggests that only the BM25 score can achieve high precision but low recall.

We achieved high performance for both of the precision and recall with $0.00 \leq \lambda_E \leq 0.50$ and $\lambda_T = 1.00$. Among these parameter values, the highest recall was achieved

Table 2 Precision for entity types for 25 combinations of λ_E and λ_T.

λ_E \ λ_T	0.00	0.25	0.50	0.75	1.00
0.00	**0.659**	0.553	0.554	0.552	0.569
0.25	0.517	0.554	0.560	0.554	0.571
0.50	0.517	0.543	0.549	0.562	0.571
0.75	0.517	0.540	0.547	0.552	0.550
1.00	0.517	0.504	0.499	0.493	0.494

Table 3 Recall for entity types for 25 combinations of λ_E and λ_T.

λ_E \ λ_T	0.00	0.25	0.50	0.75	1.00
0.00	0.260	0.443	0.445	0.441	0.456
0.25	0.198	0.448	0.452	0.446	**0.457**
0.50	0.198	0.429	0.442	0.449	0.453
0.75	0.198	0.438	0.446	0.446	0.438
1.00	0.198	0.413	0.406	0.398	0.371

Table 4 F-measure for entity types for 25 combinations of λ_E and λ_T.

λ_E \ λ_T	0.00	0.25	0.50	0.75	1.00
0.00	0.325	0.449	0.451	0.448	0.463
0.25	0.254	0.453	0.457	0.451	**0.465**
0.50	0.254	0.439	0.449	0.458	0.464
0.75	0.254	0.442	0.448	0.450	0.446
1.00	0.254	0.413	0.408	0.400	0.391

when $\lambda_I = 0.25$ and $\lambda_T = 1.00$. Moreover, we obtained the highest F-measure for $\lambda_E = 0.25$ and $\lambda_T = 1.00$ as shown in Table 4. This result suggests that we should not rely much on the similarity, but we should give a high weight to the document frequency in the Co-HITS algorithm. Note, however, that this does not mean the BM25 score is not necessarily used in our method as the BM25 score is still required to retrieve entities used in the Co-HITS algorithm. We could get more irrelevant entities if we directly found entity names from Web search results without knowing the candidate entities by the entity types, since irrelevant but common entity names may appear in the search results (*e.g.* "I" (a song title) or "IT" (a film title)).

4.2 Evaluation of Properties

This part of the evaluation was designed for evaluating the performance of our proposed property identification method. We evaluated top five properties for each modifier using: C1-C7: each of the seven criteria for identifying relevant properties, LC1-3: the method proposed in [5] that ranks entities based on the linear combination of the three criteria, SVM1-3: the SVM using the three criteria, and SVM: our proposed property identification method which we combined all seven criteria using the SVM explained in Sect. 3.3.2. The first criterion alone was used as a baseline method since the existing methods (*e.g.* [6]) applied the textual similarity between a modifier and a property in order to identify the relevant property.

Table 5 MRR for results of identifying relevant properties.

C1	C2	C3	C4	C5	C6	C7	LC1-3	SVM1-3	SVM
0.447	0.225	0.171	0.172	0.038	0.066	0.041	0.574	0.571	**0.607**

The relevance of each property was assessed by three assessors. They were required to judge a property as relevant only if a given modifier specifies a certain type of entity by property values of the property in a sequence. The inter-rater agreement was measured by Fleiss' Kappa [27] and was considered to be substantial at 0.735.

For the evaluation of each criterion (C1-C7) and LC1-3, we compared the results by Mean Reciprocal Rank (MRR), which is defined as $\frac{1}{M}\sum_{i=1}^{M}\frac{1}{r_i}$ where M is the number of modifiers in our experiment (*i.e.* 62), and r_i denotes the rank of the first relevant property for the i-th modifier. The MRR is often used for evaluating ranked lists where only the first relevant result matters and is suitable for this evaluation since we expect only a relevant property can be used for each modifier.

For the evaluation of the SVM1-3 and SVM, we extracted the results by the seven criteria using 62 modifiers with properties that already labeled (1 as relevance and 0 as irrelevance) as observed variables, and used the relevance of each property of each modifier as a predicted variable. In total, we obtained 877 samples. We utilized k-fold cross validation ($k = 5$ for the experiment) to evaluate our SVM model. In each fold, we computed and ranked the probabilities of the possible outcomes for predicted variables, *i.e.* the degree of relevance of each property regarding each modifier. Then, we measured the results by the MRR. We performed 10 rounds of k-fold cross validation to avoid the bias in the random sampling. As a result, we obtained the average MRR as the final measurement to determine the performance of our SVM model.

Experimental Results Table 5 shows the MRR achieved by seven criteria and our proposed methods for identifying relevant properties. It indicates that by using our proposed method that combines all of the seven criteria using the SVM could achieve the best performance. When comparing the performance of our proposed method (SVM) with the methods that used only the first three criteria (LC1-3 and SVM1-3), it is clear that our four new criteria contributed to our property identification. Although, their separated performances of C5, C6 and C7 are not as significant as the other four criteria.

To drill down the results in Table 5, Fig. 2 illustrates the performance in terms of the RR achieved by the first four criteria (C1-C4) and our property identification method (SVM) for each modifier, where rows represent the RR achieved by each method and columns represent the modifiers. We chose the first four criteria (C1-C4) since their MRR results suggest the highest performance among all seven in Table 5.

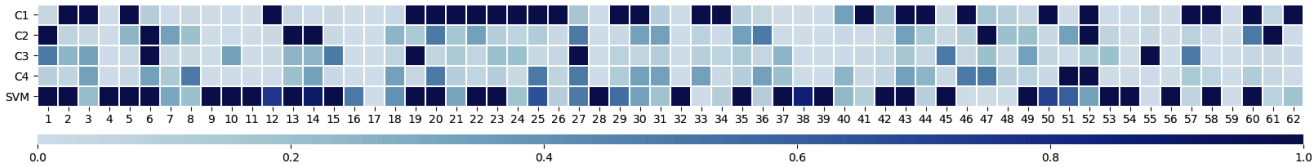


Fig. 2 Reciprocal Rank (RR) of each of the four main criteria (C1-C4) and our property identification method which applies all seven criteria (SVM) for identifying relevant properties. Color in each cell represents the degree of RR achieved for each modifier, *i.e.* 62 modifiers in total.

Table 6 Examples of properties ranked at the top by the four main criteria (C1-C4) and our property identification method (SVM). The relevant properties are the ones in red and underlined.

(a) query "ancient zen buddhist temples in kyoto"

Modifier	C1	C2	C3	C4	SVM
ancient	-	founded	mountain	venerated	mountain
	-	mountain	founded	landscape	country
	-	-	denomination	mountain	denomination
zen	denomination	school	mountain	abbot	denomination
	-	field	founded	teacher	country
	-	works	founder	landscape	lanscape
kyoto	address	school	founderpriest	residence	address
	country	residence	founder	country	country
	-	abbot	address	address	residence

(b) query "oldest architecture copenhagen"

Modifier	C1	C2	C3	C4	SVM
oldest	-	established	client	owned	constructionStartDate
	-	length	tracks	founder	line
	-	opened	architect	founded	owned

It can be seen that different criteria could identify relevant properties for different types of modifiers. For some modifiers, relevant properties could be found at high ranks by C1, while C2, C3 and C4 could not rank them near the top (*e.g.* 21st-24th modifiers from the left). On the other hand, if relevant properties could be ranked high by C2, C3, or C4, C1 could not rank them near the top in some cases (*e.g.* 13th-15th modifiers from the left). These observations suggest that the combination of the four criteria with the other three criteria, *i.e.* all seven criteria, using the SVM could identify relevant properties most accurately by complementing each other as can be seen with the results of RR for the row representing SVM.

Table 6 shows examples of relevant properties. It suggests that if the modifier corresponds to qualitative properties, their properties can be successfully found by C1. If the modifier corresponds to quantitative properties, their properties can be identified by either C2, C3, or C4. These results are also consistent with our observation in Fig. 2 that different criterion could identify relevant properties for different modifiers, and they suggest that our proposed property identification method (SVM) achieved the best performance.

4.3 Evaluation of Entity Ranking

To evaluate the performance of the proposed methods for returning a ranked listed of entities, we pooled top ten entities from six systems:

1. BM25: entities ranked by the highest BM25 score of their entity type.
2. WBER: entities ranked by the Web-based entity ranking (Eq. (2)).
3. PBER (1): entities ranked by the property-based entity ranking (Eq. (13)) using the most relevant property ($R = 1$).
4. PBER (3): entities ranked by the property-based entity ranking (Eq. (13)) using the three most relevant properties ($R = 3$).
5. WBER + PBER (1): entities ranked by the Web-based entity ranking (Eq. (2)) and property-based entity ranking (Eq. (13)) using the most relevant property ($R = 1$).
6. WBER + PBER (3): entities ranked by the Web-based entity ranking (Eq. (2)) and property-based entity ranking (Eq. (13)) using the three most relevant properties ($R = 3$).

The relevance of entities were assessed by the same

assessors as those involved in the evaluation of properties. They were required to judge the relevance of entities for each of fifty queries based on the property value of each property with the modifiers using one of the following criteria:

- **completely relevant (4)**: property values of the entity highly correspond to all modifiers and the entity also corresponds to an entity type name in the query.
- **relevant (3)**: property values of the entity correspond to all modifiers and the entity also corresponds to an entity type name in the query.
- **somewhat relevant (2)**: property values of the entity correspond to some modifiers and the entity also corresponds to an entity type name in the query.
- **not so relevant (1)**: none of property values of the entity correspond to modifiers, but the entity corresponds to an entity type name in the query.
- **completely irrelevant (0)**: the entity does not correspond to an entity type name in the query.

Note that the assessors may find additional information if the provided property values are insufficient.

We computed the nDCG [28] as it is usually used to evaluate the effectiveness of the ranking method. *nDCG@k* is defined as $\mathrm{nDCG_k} = \frac{DCG_k}{idealDCG_k}$, where k indicates the number of entities, $DCG_k = \sum_{i=1}^{k} \frac{rel_i}{log_2(i+1)}$, where i is the position of the entity in the rank and rel_i denotes the relevance of the entity at position i, and $idealDCG_k$ is the ideal DCG by the relevance of k entities.

Experimental Results Table 7 shows the performance of entity ranking in terms of the nDCG@3, 5, and 10 using 50 queries from six systems. The results suggest that entities ranked by the combination of the Web-based entity ranking and property-based entity ranking using top three relevant properties (WBER + PBER (3)) achieved the highest nDCG@3, 5 and 10, while PBER (1) achieved the lowest. For WBER, its ranking performance was good, but it is still not better than the performance of (WBER + PBER (1)) and (WBER + PBER (3)). However, WBER beats the performance of the system BM25 that simply used the similarity score between the query with the entity type names from a knowledge base since we also applied the document frequency of the entities from Web search results using the search query. Moreover, by using the property-based entity ranking, that involves the ranked list of entities by Web-based entity ranking with the relevant properties for the modifiers in the query (WBER + PBER (1) and WBER + PBER (3)), improves the ranking performance as this system explicitly concerns about the modifiers. Ranking by the relevant properties of the modifiers affects the better entity ranking results because it directly distinguishes the entities by their property values of the relevant properties (*i.e.* entity-related information), which the functionality of

Table 7 nDCG@3, 5, and 10 of six systems for returning a ranked list of entities using 50 queries.

	nDCG@3	**nDCG@5**	**nDCG@10**
BM25	0.459	0.487	0.529
WBER	0.600	0.628	0.692
PBER (1)	0.415	0.423	0.434
PBER (3)	0.537	0.541	0.580
WBER + PBER (1)	0.642	0.679	0.733
WBER + PBER (3)	**0.654**	**0.687**	**0.745**

Table 8 Examples of top ranked entities by the six systems.

(a) query "ancient zen buddhist temples in kyoto"

BM25	**Rel**	WBER	**Rel**	PBER (1)	**Rel**
Byodo-in	1.333	Kiyomizu-dera	1.667	Kennin-ji	3.333
Historical Sites of Prince Shotoku	0	Kinkaku-ji	3.0	Tenryu-ji	3.333
Sanzen-in	1.333	Ninna-ji	2.0	Kinkaku-ji	3.0
PBER (3)	**Rel**	WBER + PBER (1)	**Rel**	WBER + PBER (3)	**Rel**
Tenryu-ji	3.333	Kinkaku-ji	3.0	Kinkaku-ji	3.0
Kennin-ji	3.333	Kiyomizu-dera	1.667	Kennin-ji	3.333
Ginkaku-ji	3.0	Kennin-ji	3.333	Kiyomizu-dera	1.667

(b) query "oldest architecture copenhagen"

BM25	**Rel**	WBER	**Rel**	PBER (1)	**Rel**
Architecture in Copenhagen	0.0	Amalienborg	3.0	Tøjhus Museum	2.667
COBE Architects	0.333	Rosenborg Castle	4.0	Ny Carlsberg Glyptotek	2.333
Rørbæk & Møller Arkitekter	0.333	Christiansborg Palace	2.333	National Gallery of Denmark	2.0
PBER (3)	**Rel**	WBER + PBER (1)	**Rel**	WBER + PBER (3)	**Rel**
Henning Larsen Architects	0	Amalienborg	3.0	Amalienborg	3.0
Tegnestuen Vandkunsten	0	Rosenborg Castle	4.0	Rosenborg Castle	4.0
Tøjhus Museum	2.667	Christiansborg Palace	2.333	Christiansborg Palace	2.333

Web search engine cannot handle and process the modifiers in a search query effectively. However, we cannot use the property-based entity ranking without the Web-based entity ranking method as the performance results of PBER (1) and PBER (3) suggested. This is because the Web-based entity ranking helps finding the entities that would correspond to the entity type name in the query. Without this method, we could get the entities that relevant to the modifiers, but not relevant to the entity type name in the query.

Table 8 shows examples of entities ranked by six systems for two queries. The results suggest that WBER + PBER (3) could achieve and get the relevant entities ranked near the top of the list for both queries.

5. Conclusions

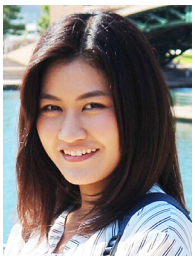
This paper proposed methods of returning a ranked list of entities for a given query by leveraging different types of modifiers. Our first proposed method, *i.e.* the Web-based entity ranking, involves the similarity between a query and entity type names together with the frequency of entities in search results returned in response to the query. Moreover, we proposed the property-based entity ranking which includes the part of identifying a property corresponding to each modifier in a query in which we proposed the property identification method based on the combination of seven different criteria using SVM. Experimental results showed that our property identification method could identify more relevant properties, and the combination of our Web-based and property-based entity ranking methods could return more relevant entities at the top rank. In the future work, we aim to find other property identification methods that can cover as many types of modifiers as possible. We also want to enhance the performance of the entity ranking methods by considering other data sources rather than Web search results and knowledge bases.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers 26700009, 18H03244 and 18H03243.

References

[1] J. Guo, G. Xu, X. Cheng, and H. Li, "Named entity recognition in query," SIGIR, pp.267–274, 2009.
[2] X. Yin and S. Shah, "Building taxonomy of Web search intents for name entity queries," WWW, pp.1001–1010, 2010.
[3] N. Phan, P. Bailey, and R. Wilkinson, "Understanding the relationship of information need specificity to search query length," SIGIR, pp.709–710, 2007.
[4] T. Lau and E. Horvitz, "Patterns of search: analyzing and modeling web query refinement," UM99 User Modeling, pp.119–128, Springer, 1999.
[5] W. Imrattanatrai, M.P. Kato, and K. Tanaka, "Entity search by leveraging attributive terms in sentential queries over rdf data," WI, pp.769–776, 2017.
[6] S. Elbassuoni and R. Blanco, "Keyword search over rdf graphs," Proceedings of the 20th ACM international conference on Information and knowledge management, pp.237–242, ACM, 2011.
[7] V. Hristidis and Y. Papakonstantinou, "Discover: Keyword search in relational databases," VLDB' 02: Proceedings of the 28th International Conference on Very Large Databases, pp.670–681, Elsevier, 2002.
[8] S. Agrawal, S. Chaudhuri, and G. Das, "Dbxplorer: A system for keyword-based search over relational databases," ICDE, pp.5–16, 2002.
[9] B. Aditya, G. Bhalotia, S. Chakrabarti, A. Hulgeri, C. Nakhe, P. Parag, and S. Sudarshan, "Banks: Browsing and keyword searching in relational databases," Proceedings of the 28th International Conference on Very Large Data Bases, VLDB' 02, pp.1083–1086, VLDB Endowment, 2002.
[10] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," tech. rep., Stanford InfoLab, 1999.
[11] L. Zou, R. Huang, H. Wang, J.X. Yu, W. He, and D. Zhao, "Natural language question answering over rdf: a graph data driven approach," SIGMOD, pp.313–324, 2014.
[12] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano, "Template-based question answering over rdf data," WWW, pp.639–648, 2012.
[13] D. Gerber and A.C.N. Ngomo, "Bootstrapping the linked data web," 1st Workshop on Web Scale Knowledge Extraction@ ISWC, 2011.
[14] A.P. De Vries, A.-M. Vercoustre, J.A. Thom, N. Craswell, and M. Lalmas, "Overview of the INEX 2007 entity ranking track," INEX, pp.245–251, 2007.
[15] G. Demartini, A.P. de Vries, T. Iofciu, and J. Zhu, "Overview of the INEX 2008 entity ranking track," INEX, pp.243–252, 2008.
[16] G. Demartini, T. Iofciu, and A.P. De Vries, "Overview of the INEX 2009 entity ranking track," INEX, pp.254–264, 2009.
[17] K. Balog, P. Serdyukov, and A.P.D. Vries, "Overview of the TREC 2010 entity track," TREC, 2010.
[18] R. Kaptein, P. Serdyukov, A. De Vries, and J. Kamps, "Entity ranking using Wikipedia as a pivot," CIKM, pp.69–78, 2010.
[19] G. Demartini, C.S. Firan, T. Iofciu, R. Krestel, and W. Nejdl, "Why finding entities in Wikipedia is difficult, sometimes," Information Retrieval, vol.13, no.5, pp.534–567, 2010.
[20] K. Balog, M. Bron, and M. De Rijke, "Query modeling for entity search based on terms, categories, and examples," ACM TOIS, vol.29, no.4, pp.22:1–22:31, 2011.
[21] M. Bron, K. Balog, and M. De Rijke, "Ranking related entities: components and analyses," CIKM, pp.1079–1088, 2010.
[22] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, and G. Attardi, "Ranking very many typed entities on wikipedia," Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp.1015–1018, ACM, 2007.
[23] J. Pehcevski, A.M. Vercoustre, and J.A. Thom, "Exploiting locality of wikipedia links in entity ranking," European Conference on Information Retrieval, pp.258–269, Springer, 2008.
[24] L. Zhang, Y. Zhang, and Y. Chen, "Summarizing highly structured documents for effective search interaction," SIGIR, pp.145–154, 2012.
[25] H. Deng, M.R. Lyu, and I. King, "A generalized Co-HITS algorithm and its application to bipartite graphs," KDD, pp.239–248, 2009.
[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.
[27] J.L. Fleiss, "Measuring nominal scale agreement among many raters.," Psychological bulletin, vol.76, no.5, pp.378–382, 1971.
[28] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," ACM Transactions on Information Systems (TOIS), vol.20, no.4, pp.422–446, 2002.



Wiradee Imrattanatrai received the B.S. degree from Mahidol University, Thailand in 2014 and the M.S. degree from Kyoto University, Japan in 2017. She has been a Ph.D. candidate of Graduate School of Informatics, Kyoto University, Japan since 2017. Her research interests include web information retrieval, entity retrieval, and data mining.

Makoto P. Kato received the B.S., M.S., and PhD degrees from Kyoto University, Japan, in 2008, 2009, and 2012, respectively. He is currently a senior lecturer at Kyoto University. His research interests include a wide range of information retrieval topics, especially, interactive information retrieval, search intent detection, and entity retrieval.

Katsumi Tanaka received the B.S., M.S., and Ph.D. degrees in Information Science fromKyoto University, Japan, in 1974, 1976 and 1981, respectively. In 1986, he joined Department of Instrumentation Engineering, Faculty of Engineering at Kobe University, Japan, as an associate professor. In 1994, he became a full professor in Faculty of Engineering, Kobe University. Since 2001, he has been a professor of Graduate School of Informatics, Kyoto University. His research interests include database theory and systems, web information retrieval, and multimedia retrieval.

Masatoshi Yoshikawa is a Professor of Graduate School of Informatics at Kyoto University. He received the Dr. Eng. degrees from Department of Information Science, Kyoto University in 1985. From April 2006, he has been a professor at Kyoto University. His current research interests include multi-user routing algorithms and services, theory and practice of privacy protection, and medical data mining. He is a member of ACM, IPSJ and IEICE.