WC3: Analyzing the style of metadata annotation among Wikipedia articles by using Wikipedia category and the DBpedia metadata database

Masaharu Yoshioka[1]

Graduate School of Information Science and Technology, Hokkaido University
N14 W9, Kita-ku, Sapporo 060-0814, Japan

Abstract. WC3 (Wikipedia Category Consistency Checker) is a system that supports the analysis of the metadata-annotation style in Wikipedia articles belonging to a particular Wikipedia category (the subcategory of "Categories by parameter") by using the DBpedia metadata database. This system aims to construct an appropriate SPARQL query to represent the category and compares the retrieved results and articles that belong to the category. In this paper, we introduce WC3 and extend the algorithm to analyze efficiently additional varieties of Wikipedia category. We also discuss the metadata-annotation quality of the Wikipedia by using WC3.

1 Introduction

Wikipedia[1] is a free, Wiki-based encyclopedia that covers a wide variety of topics. Particularly for articles about named entities (e.g., person or artifact), metadata (e.g., writer or birthplace) about those entities are usually organized in "infoboxes" displayed at the start of the articles. By extracting these items of information, the DBpedia database [1] has been constructed. Because it covers a wide variety of information about named entities, DBpedia has been used as a core element of Linked Open Data [2] and for semantic annotation [3].

Another important source of information about metadata is Wikipedia category. For example, YAGO2 [4] extracts type information from them. In the Wikipedia category structure, groups of categories can have an ancestor category such as "Categories by parameter". Most categories are then represented in a set-and-topic style (e.g., "Cities in France"), whereby an original set (e.g., "cities") is divided into smaller topic categories according to a parameter value (e.g., "France"). However, because of failures in the DBpedia metadata extraction and/or incomplete coverage in assigning appropriate Wikipedia categories to the articles, there are some articles whose metadata obtained by DBpedia are inconsistent with the metadata information based on the Wikipedia category structure.

[1] http://www.wikipedia.org/

To analyze the differences between these two information resources, we previously proposed WC3 (WC-triple: Wikipedia Category Consistency Checker)[2] based on the DBpedia metadata database [5]. This system aims to construct an appropriate SPARQL query to represent a Wikipedia category that is a subcategory of "Categories by parameter". A comparison between the queried Wikipedia articles and articles that belong to the category identifies articles that lack appropriate metadata annotation and articles that are candidates for category assignment. WC3 was a first attempt to analyze Wikipedia categories in systematic approach and can find out many inconsistent metadata annotation and/or Wikipedia category labels in the Wikipedia, mostly based on human errors and misunderstanding of the metadata annotation or Wikipedia category definition described as natural language text in the category description pages.

However, the system had a scalability problem and a lack of flexibility in constructing the SPARQL queries. In this paper, we propose an extended algorithm that uses sample articles and regular expressions for efficient and flexible SPARQL query construction based on the simple analysis of Wikipedia category strings. System performance and the consistency of the metadata annotation in Wikipedia are also addressed by applying the system to large numbers of Wikipedia categories.

The paper proceeds as follows. In Section 2, we briefly review research on the quality of the Wikipedia and DBpedia and support tools for enhancing Wikipedia contents. Section 3 introduces WC3 and proposes an extended algorithm for producing efficient and flexible SPARQL queries. In Section 4, we describe the Wikipedia category structure related to "Categories by parameter" and discuss the results of an analysis by WC3. Section 5 concludes.

2 Related Work

There have been several approaches to analyzing the quality of the Wikipedia and the DBpedia. The first approach was to check the contents of Wikipedia articles manually. Giles *et al.* [6] conducted an expert comparison between the scientific contents of Wikipedia and Encyclopedia Britannica, finding that Wikipedia had almost the same accuracy and quality as Encyclopedia Britannica. Another approach was an evaluation based on the editorial history of Wikipedia articles. Stvilla *et al.* [7] proposed a framework for information-quality assessment and confirmed that the quality of the Wikipedia can measure based on article edit history metadata, such as edit histories, discussions and vote logs. Kittur *et al.* [8] pointed out that it is important to maintain good coordination among editors to improve the quality of Wikipedia articles. Hu *et al.* [9] developed a quality measurement model for Wikipedia articles based on the quality of the editors.

With respect to the quality of the DBpedia, a common approach is to compare metadata obtained from different information resources. Mendes *et al.* [10] implemented a system, "Sieve", that supports assessment of the linked-data

[2] The link for WC3 has moved from http://wnews.ist.hokudai.ac.jp/wc3/ to http://wnews.ist.hokudai.ac.jp/wc3/old

quality. This work found that the coverage of DBpedia in different languages depends on the contents (e.g., the Portuguese DBpedia has more information about Brazilian municipalities than the English DBpedia) and a framework for integrating this information was proposed. Yoshioka *et al.* [11] proposed a framework for an automatic method of discovering links between entries in GeoNames and Wikipedia articles. During this automatic link-discovery process, the system uncovered many errors about coordinate information in Wikipedia. Another approach was based on the revision history of Wikipedia. Orlandi *et al.* [12] analyzed DBpedia according to provenance information based on the revision history [12].

This research on analyzing the quality of the Wikipedia and the DBpedia did not focus on how to support Wikipedia editors in improving the quality of Wikipedia articles. One approach to quality improvement on the academic side is entity linking. Mihalcea *et al.* [13] proposed an automatic keyword-extraction system based on Wikipedia articles. This system supported the addition of links to corresponding Wikipedia articles based on the extracted results. There have been several attempts to utilize this framework (e.g., link discovery in the English Wikipedia [14] and cross-language link discovery in Wikipedia [15]). For Wikipedia category maintenance, PetScan[3] is a simple tool for identifying candidate Wikipedia articles by manually constructing queries based on information about Wikipedia articles such as templates, links, and Wikidata. However, it is not easy to construct appropriate queries for analyzing the Wikipedia categories manually. In contrast, WC3 [5] supports the automatic construction of candidate queries based on DBpedia information.

Torres *et al.* [16] proposed a framework for selecting representative Wikipedia category paths by using DBpedia SPARQL queries and Wikipedia category information. However, this system did not aim to support Wikipedia's volunteer editors.

3 WC3

3.1 Prototype of WC3

WC3 aims to support Wikipedia's volunteer editors by checking the consistency of metadata annotation related to a given Wikipedia category [5]. This is achieved by constructing an appropriate SPARQL query to represent a set-and-topic-style category. This query is used to retrieve results from DBpedia, with the retrieved results being compared with articles that belong to the given category to check their metadata-annotation consistency.

This system analyzes all metadata for articles that belong to the target category and identifies candidate attributes for use in the SPARQL query. There are two types of candidate attributes. First, there are attributes that involve the topic-related restriction. These are selected by applying an F measure (harmonic mean of precision and recall) to articles belonging to the target category. Second,

[3] https://petscan.wmflabs.org/

there are attributes that involve the set-related restriction, which are selected by using articles that belong to the target category (e.g., “Song written by Paul McCartney”) or sibling categories of the target (e.g., “Song written by Bob Dylan”). Finally, the system checks all combinations of these candidate attributes and uses a combination with the highest F-measure to generate the SPARQL query.

A prototype version of WC3 was implemented by using the 2014 version of DBpedia data[4] as the resource description framework (RDF) database and a Wikipedia dump database (dated 2014/11/6) for finding sibling categories.

3.2 Issues with the WC3 Prototype

The prototype system can generate appropriate SPARQL queries for the Wikipedia categories whose topic is person names (e.g., “Songs written by ...” or “Films directed by ...”). However, it fails to generate appropriate queries for categories whose topic is not represented as a simple metadata value (e.g., articles belonging to “People from Tokyo” could have the metadata “Tokyo”, “Tokyo, Japan”, or “Bunkyo-ku, Tokyo” for “birthplace”).

Moreover, because the prototype system tries to use the metadata of all articles belonging to a category, it requires excessive time to analyze a category with many articles (more than 10 minutes for a category with 1,000 articles). To achieve better usability, the performance should be improved.

3.3 Proposal for a New Algorithm

To address the problems in the prototype system, we propose a new algorithm for WC3 by adopting the following approaches.

- Use of a FILTER function in SPARQL:
  To generate appropriate SPARQL queries for categories whose topic is not represented as a simple metadata value, the FILTER function finds related topic attributes to use for constructing the SPARQL query. To identify topic-related strings, sibling categories are used to exclude the shared string. For example, “1981 births” has sibling categories such as “1982 births” and “1972 births”. The topic-related string “1981” is extracted by excluding “births”, and such strings are used in generating a query via the FILTER function. Since this operation is language independent, this method can be applied analysis of DBpedia and Wikipedia in any languages.
- Generating SPARQL queries by combining topic and set restrictions:
  In the prototype system, articles in sibling categories are used to identify set-related restrictions. However, there is a computational cost and the quality of the result relies on randomly selected sibling categories. In this

[4] http://wiki.dbpedia.org/Datasets2014/

paper, metadata that have a type predicate (rdf:type[5]) or a short description (dbp:shortDescription) are selected for set-related restrictions. The final SPARQL query is generated by the highest-ranked combination of one of these set-related restrictions and a topic-related restriction, using an F-measure ranking derived from an analysis of articles belonging to the category.

– Reduction of the response time by reducing the number of SPARQL queries to an RDF database. This involves:
  - Using sample articles for categories with many articles:
    Because candidate attributes that have higher recall should have a higher recall for the sample articles, we use sample articles to identify candidate attributes for generating an appropriate query. A page size threshold (*pst*) parameter is introduced to control the size of the sampling.
  - Exclusion of candidate attributes whose recall is low:
    Because the recall of a candidate attribute is an upper bound on that of a combined-query attribute, these attributes are not used in the final query, making it unnecessary to calculate the precision and F-measure for such attributes. Therefore, we sort candidate attributes based on their recall values and use *mca* (maximum candidate attributes) to pick *mca* set-related attributes and *mca* topic-related attributes for restriction.

– Classification of errors into related subcategories and other subcategories:
  When the system analyzes a category that has articles and subcategories, there are several cases where the constructed query retrieves articles that belong to a subcategory (e.g., the SPARQL query for "People from Tokyo" would retrieve articles belonging to "Writers from Tokyo", which is a subcategory of "People from Tokyo"). To clarify the difference between errors related to subcategories and other errors, error articles are checked as to whether they belong to subcategory.

Algorithm 1 shows first half of the algorithm to select candidate attributes to construct SPARQL queries. Followings are summary of the algorithms.

lines 1-3 sets up the parameters and uses *category* as input for selecting target pages for analysis.

lines 4-8 selects at most *pst* articles from Pc for candidate generation.

lines 9-10 generates candidate substrings (Fc) for using FILTER function by identifying common shared strings among sibling categories.
$uniqueSubStringSelection(target, Sc)$ is a function to select unique string of target by excluding most common shared string among target and categories in Sc. Shared substrings between two Wikipedia categories are selected from the beginning of the string and end of the string. For example, shared

[5] In this paper, we use the abbreviations "dbo", "dbp", "dbr" and "rdf" , for "http://dbpedia.org/ontology", "http://dbpedia.org/property", "http://dbpedia.org/resource" and "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"

substrings between “1990 birhts” and “1981 births” are “19” (from the beginning) and “ births” (from the end). The system selects most common shared string between $target$ and Sc and generate topic unique strings by removing common shared string from the beginning or from the end). For example, in the case of target = “1990 births” and $Sc = \{$ “1991 births”, “2000 births”, “1940 births”, “1930 births”, “1920 births” $\}$, the most common shared string from the beginning is “19” and one from the end is “ births”. Then $uniqueSubStringSelection(target, Sc)$ returns “90 births” and “1990” by removing these shared strings from $target$.

lines 11-36 generates candidate set-related restrictions (Src), topic-related restrictions(Trc), set-related restrictions with FILTER ($SrFc$) and topic-related restrictions with FILTER ($TrFc$). [6]

lines 37-40 selects top mca candidates from each candidate group by using $|PP'_{md}|$ or $|PP'_{fc,md}|$ (number of documents that have metadata md, or fc substrings in md).

Operations related to selection of the pages (lines 4-5), candidates based on FILTER operation (lines 9-10, 20-24, 29-33), selection of candidates based on recall in sample articles ($|PP'_{md}|$ or $|PP'_{fc,md}|$) (line 37-40) are newly introduced in this paper.

Algorithm 2 shows last half of the algorithm to generate candidate SPARQL queries. Followings are summary of the algorithms.

lines 1-13 evaluates all combinations of set-related restrictions (with or without FILTER) and topic-related restrictions (with or without FILTER) using precision $p_{sc,tc}$, recall$r_{sc,tc}$, and F-measure $f_{sc,tc}$.

lines 14-15 returns pair of sc and tc with highest $f_{sc,tc}$ as candidates when SPc is not empty.

lines 16-26 evaluates all set-related restrictions (with or without FILTER) and topic-related restrictions using precision p_{stc}, recallr_{stc}, and F-measure f_{stc} and returns stc with highest f_{stc} as candidates.

Compared with previous algorithm, only combination of set-related restrictions (with or without FILTER) and topic-related restrictions (with or without FILTER) are used for evaluation.

In addition, the system can also accept a manually constructed SPARQL query for evaluating the Wikipedia category. If the user is not satisfied by the generated query, the user can modify the SPARQL query to evaluate the category. A common example of this function is to use automatic query generation for a sibling category and then to modify the topic-related restriction.

The system presents results for related articles categorized into the following four types.

Found: articles that belong to the category and are retrieved by the query.

[6] Metadata related to YAGO [4] are excluded, because it uses Wikipedia category information as a resource to extract the data.

Algorithm 1 Algorithms for WC3 (Selection of candidate metadata)

```
1: $pst \leftarrow 50$
2: $mca \leftarrow 10$
3: $Pc \leftarrow$ all articles that belong to the $category$ and that are not redirect articles
4: if $|Pc| > pst$ then
5:     $Pt \leftarrow Pc$
6: else
7:     $Pt \leftarrow$ selection of $pst$ articles from $Pt$
8: end if
9: $Sc \leftarrow$ random selection of at most 20 sibling categories of $category$
10: $Fc \leftarrow uniqueSubStringSelection(target, Sc)$
11: $Md \leftarrow$ All metadata (predicate and value pairs of RDF triple) of $Pt$ not relevant
    to category-related data [6]
12: $Src \leftarrow \emptyset$
13: $Trc \leftarrow \emptyset$
14: $SrFc \leftarrow \emptyset$
15: $TrFc \leftarrow \emptyset$
16: for all $md$ in $|Md|$ do
17:     if $predicate(md)$ ==<rdf:type> || <dbp:shortDescription> then
18:         $Src \leftarrow Src \cup \{md\}$
19:         $PP'_{md} \leftarrow$ article set that has metadata $md$ in $Pt$
20:         for all $fc$ in $|Fc|$ do
21:             if $value(md)$ contains $fc$ then
22:                 $SrFc \leftarrow SrFc \cup \{\{fc, md\}\}$
23:                 $PP'_{fc,md} \leftarrow$ article set that has metadata $md$ that contains $fc$ as substring
                    in $Pt$
24:             end if
25:         end for
26:     else
27:         $Trc \leftarrow Trc \cup \{md\}$
28:         $PP'_{md} \leftarrow$ article set that has metadata $md$ in $Pt$
29:         for all $fc$ in $|Fc|$ do
30:             if $value(md)$ contains $fc$ then
31:                 $TrFc \leftarrow TrFc \cup \{\{fc, md\}\}$
32:                 $PP'_{fc,md} \leftarrow$ article set that has metadata $md$ that contains $fc$ as substring
                    in $Pt$
33:             end if
34:         end for
35:     end if
36: end for
37: $Src \leftarrow$ Sort $Src$ in descendant order by using $|PP'_{md}|$ and select top $mca$ elements.
38: $Trc \leftarrow$ Sort $Trc$ in descendant order by using $|PP'_{md}|$ and select top $mca$ elements.
39: $SrFc \leftarrow$ Sort $SrFc$ in descendant order by using $|PP'_{fc,md}|$ and select top $mca$
    elements.
40: $TrFc \leftarrow$ Sort $TrFc$ in descendant order by using $|PP'_{fc,md}|$ and select top $mca$
    elements.
```

Algorithm 2 Algorithms for WC3 (Generation of candidate SPARQL queries)

```
1: $SPc \leftarrow \emptyset$
2: for all $sc$ in $Src \cup SrFc$ do
3:   for all $tc$ in $Trc \cup TrFc$ do
4:     $PP_{sc,tc} \leftarrow$ articles that have $sc$ and $tc$ metadata in *category*
5:     if $|PP_{sc,tc}| > 0$ then
6:       $SPc \leftarrow SPc \cup sc, tc$
7:       $PA_{sc,tc} \leftarrow$ articles that contain the relevant attribute in all Wikipedia articles.
8:       $p_{sc,tc} \leftarrow |PP_{sc,tc}|/|PA_{sc,tc}|$
9:       $r_{sc,tc} \leftarrow |PP_{sc,tc}|/|Pc|$
10:      $f_{sc,tc} \leftarrow 2p_{sc,tc}r_{sc,tc}/(p_{sc,tc} + r_{sc,tc})$
11:    end if
12:  end for
13: end for
14: if $|SPc| > 0$ then
15:   return SPARQL query that uses $\{sc, tc\}$ with highest $f_{sc,tc}$ as constraints
16: else
17:   for all $stc$ in $Src \cup SrFc \cup Trc \cup TrFc$ do
18:     $PP_{stc} \leftarrow$ articles that contain the relevant attribute in *category*
19:     $SPc \leftarrow SPc \cup stc$
20:     $PA_{stc} \leftarrow$ articles that contain the relevant attribute in all Wikipedia articles.
21:     $p_{stc} \leftarrow |PP_{stc}|/|PA_{stc}|$
22:     $r_{stc} \leftarrow |PP_{stc}|/|Pc|$
23:     $f_{stc} \leftarrow 2p_{stc}r_{stc}/(p_{stc} + r_{stc})$
24:   end for
25: end if
26: return SPARQL query that uses $stc$ with highest $f_{stc}$ as constraints
```

NotFound: articles that belong to the category but cannot be retrieved by the query.

Error(SubCategory): articles that belong to a subcategory of the target category but are retrieved by the query.

Error(Other): articles that do not belong to the category or its subcategories but are retrieved by the query.

4 Wikipedia Category Analysis Using WC3

4.1 Candidate Wikipedia Categories

For this analysis, we used DBpedia (2015-04 version)[7] and the Wikipedia dump database used for DBpedia (dated 2015/04/03).

WC3 aims to analyze set-and-topic-style Wikipedia categories. In the Wikipedia category structure, those categories are located as subcategories of "Categories by parameter". However, not all subcategories of "Categories by parameter" are set-and-topic-style Wikipedia categories. For example, Wikipedia category "Hokkaido University" is a subcategory of "Universities and colleges in Sapporo", but this category is a topic category and it is difficult to analyze by using WC3. Because most of these topic categories have articles with the same name and most set-and-topic-style categories do not have such articles, we exclude categories that have articles with the same name from candidates of set-and-topic-style categories.

To discuss the applicability of WC3, we now consider the proportion of categories found as set-and-topic-style categories from subcategories of "Categories by parameter". The total number of Wikipedia categories that have at least one category page or subcategory is 1,251,889. In addition, there are a number of categories classified as "stubs". Because "stubs" categories indicate that articles are under development, it is inappropriate to analyze them by using WC3-like tools. After excluding such "stubs" categories, Wikipedia has 1,238,392 categories in total. From these categories, we found 691,671 set-and-topic-style categories and 565,431 categories with at least one article in such a category. From these results, we estimate that WC3 may help about half (565,431/1,238,392 = 45.7%) of the categories in Wikipedia.

4.2 System Implementation and Examples

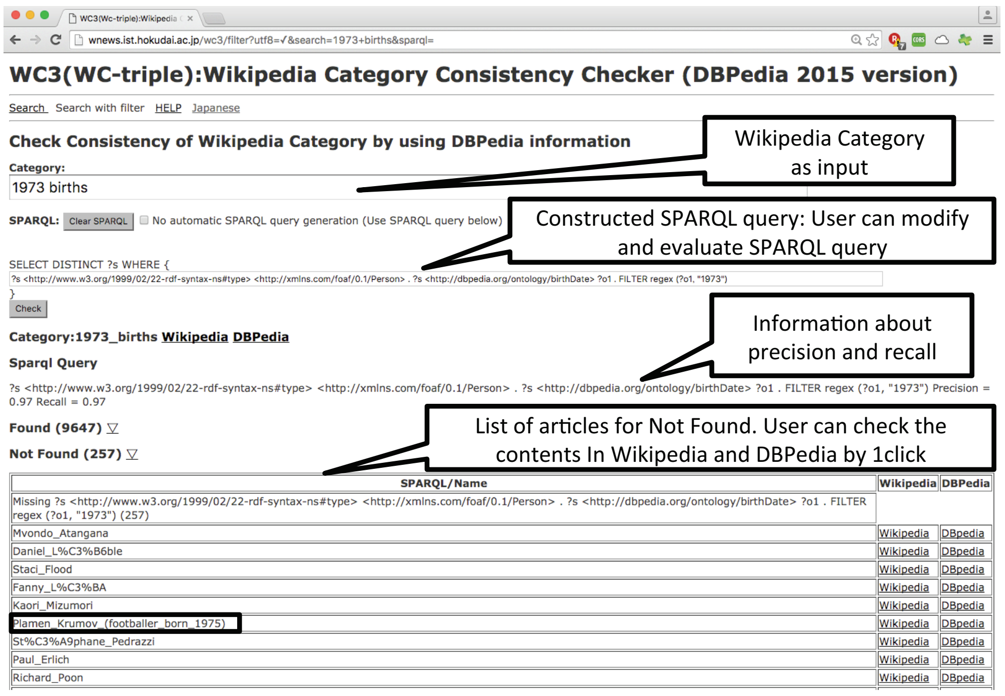
We have implemented WC3 (FILTER version)[8], which is an extended version of WC3. The parameter values used for the system were $pst = 50$ and $mca = 10$.

Figure 1 shows a screenshot of WC3 (FILTER version) and an example of the system output for the "1973 births" category.

The system constructed the following SPARQL query and found 9,647, 257, and 298 articles categorized as Found, NotFound, and Error, respectively (recall = 9,647/(9,647+257) = 0.97 and precision = 9,647/(9,647+298) = 0.97).

[7] http://wiki.dbpedia.org/Downloads2015-04

[8] http://wnews.ist.hokudai.ac.jp/wc3/


WC3(Wc-triple):Wikipedia

wnews.ist.hokudai.ac.jp/wc3/filter?utf8=✓&search=1973+births&sparql=

WC3(WC-triple):Wikipedia Category Consistency Checker (DBPedia 2015 version)

Search Search with filter HELP Japanese

Check Consistency of Wikipedia Category by using DBPedia information

Category:
1973 births

SPARQL: Clear SPARQL ☐ No automatic SPARQL query generation (Use SPARQL query below)

```
SELECT DISTINCT ?s WHERE {
?s <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> . ?s <http://dbpedia.org/ontology/birthDate> ?o1 . FILTER regex (?o1, "1973")
}
```

Check

Category:1973_births Wikipedia DBPedia

Sparql Query

?s <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> . ?s <http://dbpedia.org/ontology/birthDate> ?o1 . FILTER regex (?o1, "1973") Precision = 0.97 Recall = 0.97

Found (9647) ▽

Not Found (257) ▽

SPARQL/Name	Wikipedia	DBPedia
Missing ?s <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> . ?s <http://dbpedia.org/ontology/birthDate> ?o1 . FILTER regex (?o1, "1973") (257)		
Mvondo_Atangana	Wikipedia	DBpedia
Daniel_L%C3%B6ble	Wikipedia	DBpedia
Staci_Flood	Wikipedia	DBpedia
Fanny_L%C3%BA	Wikipedia	DBpedia
Kaori_Mizumori	Wikipedia	DBpedia
Plamen_Krumov_(footballer_born_1975)	Wikipedia	DBpedia
St%C3%A9phane_Pedrazzi	Wikipedia	DBpedia
Paul_Erlich	Wikipedia	DBpedia
Richard_Poon	Wikipedia	DBpedia

Wikipedia Category as input
Constructed SPARQL query: User can modify and evaluate SPARQL query
Information about precision and recall
List of articles for Not Found. User can check the contents In Wikipedia and DBPedia by 1click

Fig. 1. Screenshot of WC3 (FILTER version)

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type http://xmlns.com/foaf/0.1/Person .
?s dbo:birthDate ?o1 .
FILTER regex (?o1, ‘‘1973’’)
}
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

When checking the NotFound articles, there were two problematic cases. The first arises from a lack of information from the infobox (e.g., “Bai Xiaoyun”[9]). The second arises from inconsistent annotations for the infobox and Wikipedia categories (e.g., “Plamen Krumov (footballer, born 1975)”). Most of the Error articles are candidates for the category except articles that have multiple instances of birthDate information. There are two types of such articles. First, articles can involve more than one person (e.g., “List of Playboy Playmates of 1994”). Second, there can be problems with the metadata extraction in DBpedia (e.g., “Anjali Sudhakar” and “Barbara Kanam”). For the latter case, most of those articles have another birth-related Wikipedia category (e.g., “1972 births” for

[9] All English Wikipedia articles referred to in this section were accessed on April 25, 2016

"Anjali Sudhakar" and "1970 births" for "Barbara Kanam"), but there are cases where the annotated Wikipedia category seems to be wrong ("Anjali Sudhakar") and the DBpedia metadata seems to be wrong ("Barbara Kanam"). In addition, there are many articles whose metadata are inconsistent with Wikipedia categories (e.g., "Ratish Nanda" being categorized as "1974 births").

To investigate the effectiveness of system performance when using samples, we conducted experiments for "1901 births" to "2000 births" (the average number of articles for each of these categories was 7606.99). The system can generate a SPARQL query that is a simple replacement of the FILTER argument from "1973" to the year related to the category (e.g., "1900" for "1900 births"). However there are several cases that use the almost equivalent class dbo:Person, <http://schema.org/Person>, and the related class <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#Agent> instead of <http://xmlns.com/foaf/0.1/Person> for the set restriction. In these cases there are several articles that have related metadata (e.g., dbo:Person, and <http://schema.org/Person>) and do not have <http://xmlns.com/foaf/0.1/Person>. As a result recall of the query that uses <http://schema.org/Person> is lower than one that uses other related metadata. For the topic restriction, the almost equivalent attribute dbp:dateOfBirth is used in the FILTER operation. The averages for precision, recall, and F-measure for these 100 categories are 0.97, 0.96, and 0.97, respectively. The average response time for the SPARQL query-generation process[10] was 43.5 seconds. The averages for precision, recall, and F-measure for these 100 categories are 0.97, 0.96, and 0.97, respectively. The average response time for the SPARQL query-generation process[11] was 43.5 seconds.

For comparison, we also analyzed same Wikipedia categories by using algorithm of previous prototype system. The averages for precision, recall, and F-measure for these 100 categories are 0.97, 0.86, and 0.91 respectively. In these cases, the system uses dbo:birthYear instead of dbp:birthDate. However since there are several articles that have dbp:birthDate and do not have dbp:birthYear, recall of the previous system is worse than one of the proposed system. The average response time for the SPARQL query-generation was increased to 849 seconds because of checking all metadata of the articles belongs to the category. Using sample articles significantly reduce the time of SPARQL query generation for Wikipedia categories with large number of articles.

To evaluate the quality of the generated SPARQL queries, we applied WC3 to the sample of set-and-topic-style Wikipedia categories. We sorted the candidate set-and-topic-style categories based on the number of articles belonging to the category that were not redirects and used the top 5,000 for evaluation. To investigate the appropriateness of using FILTER in the analysis, we conducted experiments that compared the proposed system with a non-FILTER system.

[10] Because the rendering time for displaying the results was less than a second, the time for SPARQL query generation is almost equivalent to the total response time.

[11] Because the rendering time for displaying the results was less than a second, the time for SPARQL query generation is almost equivalent to the total response time.

For this experiment, because articles belonging to the Error(SubCategory) may be candidate articles when the subcategories are not divided into more detailed subcategories, $prec' = |PP|/(|PA| - |PS|)$, where PP, PA, and PS represent articles that satisfy the SPARQL query in the the Wikipedia category, all Wikipedia data, and in subcategories of the Wikipedia category, respectively.

Tables 1 and 2 are cross tables of recall and precision ($prec'$) for the SPARQL queries constructed by the proposed system and the system without the FILTER operation. The averages of precision and recall for the proposed system were 0.497 and 0.575, respectively, and those for the system without the FILTER operation were 0.456 and 0.540. We can confirm that there is a statistically significant difference between these two systems in terms of the Wilcoxon's signed rank test (p < 0.01).

Table 1. Cross table of recall and precision for SPARQL queries constructed by the proposed system

Precision Recall	[0,0.2)	[0.2,0.4)	[0.4,0.6)	[0.6,0.8)	[0.8,1]	Total
[0,0.2)	605	269	176	79	46	1175
[0.2,0.4)	150	212	183	71	31	647
[0.4,0.6)	115	173	164	125	51	628
[0.6,0.8)	118	138	120	97	52	525
[0.8,1)	153	192	273	415	992	2025
Total	1141	984	916	787	1172	5000

Table 2. Cross table of recall and precision for SPARQL queries constructed by the system without FILTER

Precision Recall	[0,0.2)	[0.2,0.4)	[0.4,0.6)	[0.6,0.8)	[0.8,1]	Total
[0,0.2)	777	335	177	74	51	1414
[0.2,0.4)	161	230	139	57	31	618
[0.4,0.6)	139	181	138	110	44	612
[0.6,0.8)	130	125	100	94	39	488
[0.8,1)	157	204	250	331	926	1868
Total	1175	1049	923	807	1046	5000

From these tables, we can confirm that WC3 with a FILTER operation can generate SPARQL queries that are more appropriate for representing Wikipedia categories. However, the performance of the system is inadequate for analyzing all of those Wikipedia categories. In analyzing the success and failure of the system, the following examples are used in the discussion.

– Higher recall and precision:
  "Films directed by D. W. Griffith" (prec = 0.99, recall = 0.95) is an example of using a combination of set and topic restrictions.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type dbo:Wikidata:Q11424 .
?s dbo:director ?o1 .
FILTER regex (?o1, ‘‘D._W._Griffith’’)
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

  "Portugal international footballers"(prec = 0.99, recall = 0.97) is an example of when using a corresponding property to represent topic restriction is good enough.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type http://xmlns.com/foaf/0.1/Person .
?s dbp:nationalteam dbr:Portugal_national_football_team .
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

  In both cases, Error(Other) contains candidate articles for adding the target Wikipedia category.

– Higher precision with modest recall:
  "University of Michigan alumni" (prec = 0.98, recall = 0.31) is an example of using a combination of set and topic restrictions.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type http://xmlns.com/foaf/0.1/Person .
?s dbp:almaMater ?o1 .
FILTER regex (?o1, "University_of_Michigan")
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

  "Hungarian canoeist" (prec = 1.0, recall = 0.67) is an example of using a corresponding shortDescription if one exists. Recall of the query relies on the one for extracting such metadata from the articles.

```
SELECT DISTINCT ?s
WHERE {
?s <dbp:shortDescription Hungarian canoeist> .
?s <http://purl.org/dc/elements/1.1/description Hungarian canoeist> .
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

  For "University of Michigan alumni", Error(Other) contains candidates for adding the target. In both cases, NotFound contains a list of articles without common metadata annotation.

– Higher recall with modest precision:
  "American metalcore musical groups"(prec = 0.44 and recall = 0.75) is an example of requiring additional topic restrictions.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type dbo:Band .
?s dbo:genre dbr:Metalcore
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

"Public high schools in North Carolina"(prec=0.42 and recall = 0.78) is an example of lacking an appropriate set restriction.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type dbo:School .
?s dbo:city ?o1 .
FILTER regex (?o1, "North_Carolina")
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

In the former case, precision would be improved by adding another restriction to represent "American".

– Modest recall and modest precision:
  "People from Tokyo" (prec = 0.53, recall = 0.59) is an example that shows incomplete coverage of articles for a particular Wikipedia category.

```
SELECT DISTINCT ?s
WHERE {
?s rdf:type http://xmlns.com/foaf/0.1/Person .
?s dbp:placeOfBirth ?o1 .
FILTER regex (?o1, ‘‘Tokyo’’)
MINUS { ?s dbo:wikiPageRedirects ?o . }}
```

  Because "People from Tokyo" is a category for "people who were born in or who are residents of Tokyo, Japan," [12] the generated SPARQL query seems to be reasonable for selecting people who were born in Tokyo. Low precision means the incompleteness of the Wikipedia category as an index for articles. Therefore, Error(Other) may contain candidates for adding the target. However, this query does not find people who were residents of Tokyo.
– Lower recall and precision:
  The system cannot generate appropriate SPARQL queries for categories whose topics are unknown (e.g.,"Date of birth unknown"). Moreover, the system can fail to construct an appropriate query for multiple topic restrictions (e.g., "American people of Swedish descent" and "Expatriate soccer players in South Africa"). This can be fixed by adding another topic restriction in some cases, but there are several cases for which the vocabulary of DBpedia is insufficient to represent the differences (e.g., the distinction between "English female singers" and "English male singers" is difficult because there is no clear attribute for gender). Another type of failure relates to the limitations of the constructed SPARQL queries (e.g., "20th-century

[12] https://en.wikipedia.org/wiki/Category:People_from_Tokyo

births" and "S.League players"). For the category "20th-century births", it might be better to use the regular expression "19??" for dateOfBirth, but this system cannot generate such an expression. For "S.League players", it is necessary to represent the relationship between "S.League" and "team" in constructing an appropriate query.

4.3 Discussion

From this analysis, we can confirm that the new WC3 (FILTER version) can construct SPARQL queries that are more appropriate than those constructed by the previous prototype system. In addition, the use of sample articles reduced the response time.

However, we can identify three issues to be considered for improving the function of the system.

- Support for constructing more flexible SPARQL queries:
  As discussed using the example of lower recall and precision, it is necessary to adopt another strategy for constructing SPARQL queries involving the use of regular expression patterns for particular types of topic (e.g., "19??" for "20th-century") and the use of a part–whole relationship (e.g., relationships between "team" and "league" or between "region" and "subregion").
  In addition to such automatic support, we also plan to make a database of SPARQL queries for representing Wikipedia category that were confirmed manually. The voluntary editors can use such query when it exists. Even if there is no corresponding query for the category, query of sibling categories that have different topic related restriction may be helpful to construct new SPARQL query by replacing topic related restriction(s) in the query.
- Feedback for DBpedia metadata extraction:
  As discussed using the example of "1973 births", there are several cases where the extracted DBpedia metadata themselves are inconsistent (e.g., multiple birthDates for one person). It would be better to have a framework for identifying such problems, which would improve the quality of metadata extraction in DBpedia.
- Framework to provide feedback information when Wikipedia's volunteer editors update Wikipedia articles:
  One of the problems of the system is using fixed DBpedia data. It is helpful to discover problems that exist at that time. However, an editor cannot check if a particular update would adequately address the problem. For such problems, it would be preferable to have a framework for updating the DBpedia database based on editor-supplied updates. For example, when a volunteer editor checks a Wikipedia category, the editor could request updates for the metadata of articles that belong to that category. We have already started to discuss this issue with the Japanese DBpedia community[13] for the Japanese version of WC3, called WC3ja[14].

[13] http://ja.dbpedia.org/
[14] http://wnews.ist.hokudai.ac.jp/wc3ja

5 Conclusions

In this paper, we have proposed an extension of WC3 that uses a FILTER function to represent SPARQL queries for Wikipedia categories whose common metadata values are not just simple values. In addition, we have investigated a sampling approach for identifying candidate attributes that is both adequate for the construction of appropriate SPARQL queries and that improves the response time.

This approach has also clarified the situation where editors aiming to maintain an infobox may overlook the addition of appropriate Wikipedia categories. As a result, there are several Wikipedia categories whose coverage of related articles is incomplete.

Even though there remain several issues with the new system, providing the system to Wikipedia's volunteer editors would help improve the quality of metadata annotation in Wikipedia and, as a result, the quality of the DBpedia would also improve.

Acknowledgement

This work was partially supported by JSPS KAKENHI Grant Number 25280035 and 16H01756.

References

1. 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** (2009) 154 – 165
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. International Journal on Semantic Web and Information Systems **5** (2009) 1–22
3. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: Shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems. I-Semantics '11, New York, NY, USA, ACM (2011) 1–8
4. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: A spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence **194** (2013) 28 – 61
5. Yoshioka, M., Loban, R.: WC3: Wikipedia Category Consistency Checker based on DBPedia. In: Proceedings of 11th International Conference on Signal-Image Technology & Internet-Based Systems. (2015) 712–718
6. Giles, J.: Internet encyclopaedias go head to head. Nature **438** (2005) 900–901
7. Stvilia, B., Gasser, L., Twidale, M.B., Smith, L.C.: A framework for information quality assessment. Journal of the American Society for Information Science and Technology **58** (2007) 1720–1733
8. Kittur, A., Kraut, R.E.: Harnessing the wisdom of crowds in Wikipedia: Quality through coordination. In: Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work. CSCW '08, New York, NY, USA, ACM (2008) 37–46

9. Hu, M., Lim, E.P., Sun, A., Lauw, H.W., Vuong, B.Q.: Measuring article quality in Wikipedia: Models and evaluation. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management. CIKM '07, New York, NY, USA, ACM (2007) 243–252
10. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: Linked data quality assessment and fusion. In: Proceedings of the 2012 Joint EDBT/ICDT Workshops. EDBT-ICDT '12, New York, NY, USA, ACM (2012) 116–123
11. Yoshioka, M., Kando, N.: Issues for linking geographical open data of GeoNames and Wikipedia. In Takeda, H., Qu, Y., Mizoguchi, R., Kitamura, Y., eds.: Semantic Technology. Volume 7774 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 375–381
12. Orlandi, F., Passant, A.: Modelling provenance of DBpedia resources using Wikipedia contributions. Web Semantics: Science, Services and Agents on the World Wide Web **9** (2011) 149 – 164 Provenance in the Semantic Web.
13. Mihalcea, R., Csomai, A.: Wikify!: Linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management. CIKM '07, New York, NY, USA, ACM (2007) 233–242
14. Xu, M., Wang, Z., Bie, R., Li, J., Zheng, C., Ke, W., Zhou, M.: Discovering Missing Semantic Relations between Entities in Wikipedia. In: The Semantic Web – ISWC 2013: 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I. Springer Berlin Heidelberg, Berlin, Heidelberg (2013) 673–686
15. Tang, L.X., Kang, I.S., Kimura, F., Lee, Y.H., Trotman, A., Geva, S., Xu, Y.: Overview of the NTCIR-10 cross-lingual link discovery task. In: Proceedings of the 10th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Quesiton Answering, And Cross-Lingual Information Access. (2013) 8–38
16. Torres, D., Molli, P., Skaf-Molli, H., Diaz, A.: Improving Wikipedia with DBpedia. In: Proceedings of the 21st International Conference on World Wide Web. WWW '12 Companion, New York, NY, USA, ACM (2012) 1107–1112