Knowledge graph matching with inter-service information transfer

Semih Yumusak[1,2][0000−0002−8878−4991]

[1] KTO Karatay University, Turkey semih.yumusak@karatay.edu.tr
[2] AI4BD AG Switzerland semih.yumusak@ai4bd.com

Abstract. Knowledge graph matching is an approach to create entity mappings of structured data with linked data sources. As an automated approach, this paper explains a sparql query based matching engine developed during the Columns-Property Annotation (CPA) Challenge under ISWC 2020 SemTab challenge (Semantic Web Challenge on Tabular Data to Knowledge Graph Matching). The proposed approach utilizes a text correction via different knowledge base services/libraries as well as numeric interval definitions to identify negligible numeric differences. The approach (submitted as TeamTR) achived, in the CPA task, F1-scores of 0.916. 0.873 and 0.837 in Rounds 1, 2 and 3, respectively.

Keywords: Knowledge Graph Matching · Knowledge Transfer · Text recommendation · SemTab Challenge

1 Introduction

Knowledge graphs are graph structured databases, which are defined with ontology and vocabulary definitions. The graph structure standards form the basis of the Semantic Web [1]. Whereas one may create local knowledge graph structures for a specific domain, knowledge graphs may broadly contain cross-domain, generic, or domain specific knowledge. Knowledge bases may be published as open data, which is called as linked open data [2]. As one of the largest cross-domain knowledge graph, Wikidata [4] is a web project to semantically annotate Wikipedia free web encyclopedia. In the SemTab challenge [5], tabular extractions of specific Wikidata entities are provided in a comma delimited format. The aim of the challenge is to match the related Wikidata entities and properties back to their Wikidata URIs. There are three sub challanges defined under SemTab, which are; Column-Type Annotation (CTA) Challenge, Cell-Entity Annotation (CEA) Challenge, and Columns-Property Annotation (CPA) Challenge. This study focuses on the Columns-Property Annotation (CPA) Challenge, which aims at "Assigning a KG property to the relationship between two columns" [3]. In the following sections, the challenge and the input dataset is described. Then, the column property annotation methodology is explained with a workflow diagram. Finally, the results and recommended strategies are discussed.

2 Challenge and Dataset Definition

The CPA challenge provides tables containing a list of records with the same type and relational mappings. In Table 1, a sample input table is provided. The first column contains an entity label, whereas the other columns contain related attributes. The aim of this challenge is to identify what the relation between column 0 and the rest of the columns is, more generally is to identify the relationship among two columns. For instance, from the first row of Table 1 ”Leesmuseum” is a society, which is `located in` ”Amsterdam/Netherlands” and which has an `inception date` as 1800-11-17, and additionally an `instance of` a reading museum.

Table 1. Example Wikidata Extracted Input Table

col0	**col1**	**col2**	**col3**	**col4**
Leesmuseum	Amsterdam	Netherlands	1800-11-17	reading museum
The Marlowe	Cambridge	United Kingdom	1.05.1907	theatrical troupe
Club Gorca	Seville	Spain	1.01.1966	organization

3 Column Property Annotation Methodology

In order to get the Wikidata type of the related properties (i.e. attribute definitions listed in col1...coln) for the main entity (col0), sequential operations were performed for each entity. In Figure 1, the simple process flow is illustrated, which was used in Round 1.

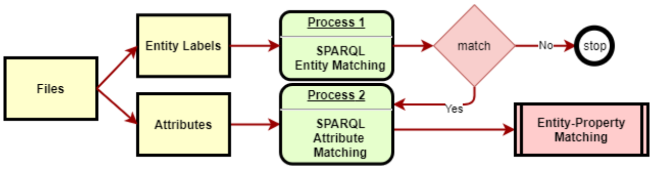

Fig. 1. Matching process for TeamTR Round 1 Solution

The process starts by creating a simple exact matching query (See Process 1 in Figure 1). Within the same query, the attribute labels matching with the entity are additionally collected (See Process 2 in Figure 1). Below is the SPARQL query to select both the entity and the matching attributes.

```
SELECT * WHERE  {
     ?s rdfs:label [col0].
     ?s ?p2 ?o.
     ?o ?p3 [colx]. }
```

The process achieved an F1-score of 0.916 by the challenge submission system in the first round. The query may find only the specific attribute and reversely retrieve its predicate. However, this simplistic approach is only limited to exact property matching even if the entity definition is present. While doing that exact mapping, there existed a simple type matching for numeric and date types. Such as;

- if the value is a string, *@en* is added as a suffix
- if the value is numeric, the value is kept as it is
- if the value is date a suffix of *T00:00:00Zxsd:dateTime* was added in order to query date types from the knowledge base.

The preliminary exact property matching approach was a very limited approach in terms of query performance and the results. The sequential operations were improved in the second round. Spell checking (See Process 3 in Figure 2) was additionally performed to correct misspelled words and a wikidata search (See Process 4 in Figure 2) is utilized to text search for partial strings within Wikidata.

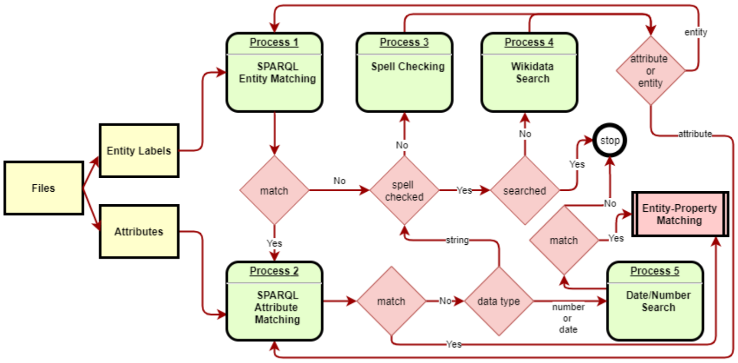

Fig. 2. Matching process for TeamTR Round 2 Solution

Within the second round, the date and number search was additionally separated from the simple entity matching query. To utilize that approach, first, entity URI matching was performed to retrieve the entity URI, then the related entities with their properties extracted. The literal properties of an entity may either exist at the first level neighborhood or the second level neighborhood. Thus, two different SPARQL queries were generated to extract both first level and second level neighbors with their properties. The following SPARQL queries were used to retrieve attributes of a matched entity.

```
SELECT * WHERE { [entity\_URI] ?p0 ?o0 .}
or
SELECT * WHERE \{ [entity\_URI] ?p0 ?o0 . ?o0 ?p1 ?o2.}
```

Within the selected attributes, filtering was applied based on the values of the input table. For the numeric data types, a threshold value was set (in percentages) to select the closest value compared to the input value. For date and text inputs, exact matching was required.The following SPARQL filters were applied to select properties for numbers, date, and text values.

```
FILTER (?o> [colx-threshold]  && ?o< [colx+threshold])
or
FILTER (?o= colx)
```

The second round solution achieved an F1-score of 0.873 by the challenge submission system.

In the third round, it was realized that there were significant amounts of spelling errors, numeric differences in the input data. Thus, the matching process was improved by adding the following functionalities. First, spell checking for the incorrectly spelled phrases was utilized by approximate string matching. However, string similarity based approaches may come up with erroneous replacements. In order to overcome incorrect replacements, context-aware spell checking was finally utilized (See Process 6 in Figure 3) as a more advanced method to take into account the context of the phrase. As a context-aware knowledge base, search engines were used to correct the misspelled phrases with their correct replacements in process 5. The phrases were submitted to search engines such as google and yandex. From the search results, automated corrections of the phrases were extracted which were using the search engine's knowledge base at the backend. For instance, whether you search for the phrase "linked data", or an incorrectly spelled "linked dta" phrase; a search engine shows corrected results which has the context around semantic web. By using this search engine powered phrase correction, the misspelled entity label strings was corrected and retrieved from Wikidata.

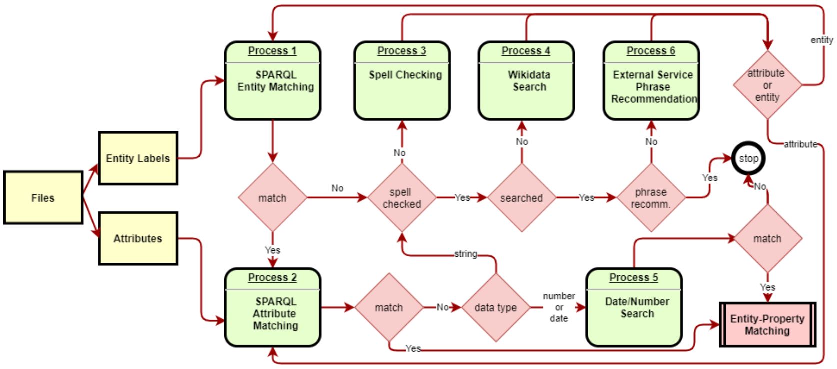

Fig. 3. Matching process for TeamTR Round 3 Solution

The complete flow for the correction and matching processes are illustrated in Figure 3. The third round solution achieved an F1-score of 0.837 by the challenge submission system.

4 Conclusion and Future Work

Within the SemTab challenge, this study proposes a series of deterministic approaches to match entities and their properties. In case of misspellings, the system proposes to use external services such as search engines to correct words or phrases. While searching for the possible entity property matching, numerical proximity was used to identify numeric properties. The automated approach achieved an F1-score of 0.837 with no manual corrections or adoptions on the result set.

References

1. Berners-Lee, Tim and Hendler, James and Lassila, Ora: The semantic web. Scientific American **284**(5), 34–43 (2001)
2. The Linked Open Data Cloud, https://lod-cloud.net/. Last accessed Nov. 1, 2020
3. Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen and Kavitha Srinivas : SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. Extended Semantic Web Conference (ESWC). 2020.
4. Vrandečić, Denny and Krötzsch, Markus: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
5. SemTab 2020: Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, https://www.cs.ox.ac.uk/isg/challenges/sem-tab/2020/index.html. Last accessed Nov. 1, 2020