Matching CSV to Wikidata via Meta-lookup

Renat Shigapov, Philipp Zumstein, Jan Kamlah, Lars Oberländer, Jörg Mechnich and Irene Schumm

Mannheim University Library, University of Mannheim
Business and Economics Research Data Center Baden-Württemberg
SEMTAB2020: ENTITY, PROPERTY AND TYPE ANNOTATIONS

**INPUT**

<table>
<thead>
<tr>
<th>col0</th>
<th>col1</th>
<th>col2</th>
</tr>
</thead>
<tbody>
<tr>
<td>watt</td>
<td>power</td>
<td>1</td>
</tr>
<tr>
<td>hertz</td>
<td>frequency</td>
<td>1</td>
</tr>
</tbody>
</table>

**OUTPUT**

- **coherent SI unit**: watt
- **measured quantity**: power
- **quantity**: 1

- **coherent SI unit**: hertz
- **measured quantity**: frequency
- **quantity**: 1

- **rdfs:label**: measured physical quantity
- **conversion to SI unit**:

---

1R7JYOUO.csv in round 4
KEA IDEAS: META-LOOKUP AND CONTEXTUAL MATCHING

**Meta-lookup**
lookup over many search and metasearch engines

**Contextual matching**
matching an item using the at least two features

- hertz + frequency = hertz
- herzt + coherent SI unit = hertz
- herzt + P111 frequency = hertz
- herzt + P2370 frequency = hertz
2.3 Workflow: 7 steps

We implemented \texttt{bbw} as the 7-steps workflow illustrated by Fig. 3.

Step 1: Preprocessing.
We fix encoding and clean a text using \texttt{ftfy} library\footnote{https://docs.python.org/3/library/re.html}. We predict four datatypes (number, time, name and string) using \texttt{re} module.

Step 2: Meta-lookup a subject’s label & match the object’s labels.
The meta-lookup returns the labels of candidate Items. The Wikidata SPARQL endpoint returns all triples, which have the objects equal to the labels of the candidate Items. The Properties and types are also returned. We match the values in the tail cells. If matching successful, we take the subject’s IRI, the subject’s type and the Property. If the object is an Item, we take the object’s IRI and the object’s type(s) as well.

Step 3: Lookup the object’s labels with their properties & match a subject’s label and the object’s labels.
We query the Wikidata SPARQL endpoint with the values of the tail cells and their properties inferred from Step 2. The endpoint returns the subject’s labels and the types of subjects and objects. We match the value in the first cell to the subject’s label using the edit distance. Then, we match the values in the tail cells as in Step 1.

Step 4: Lookup the object’s labels & match a subject’s label and the object’s labels.
In contrast to Step 3, we do not use Properties in Step 4. We query the Wikidata SPARQL endpoint with the values of the tail cells, which are entities. It results in the subject’s labels and the types of a subject and the objects. We match the cell values as in Step 3.

Step 5: Lookup the object’s labels with its type.
We query the Wikidata SPARQL endpoint using the value of a tail (entity) cell and its type inferred in previous Steps. We take the object’s IRI.

Step 6: Lookup all Items with a certain type & match the object’s labels.
The Wikidata SPARQL endpoint returns all Items with the type inferred in previous Steps. We match the subject’s label with the edit distance. We send a SPARQL-query again and receive all triples, which have the objects equal to the labels of the candidate Items. We match the values in the tail columns as in Step 1.

Step 7: Postprocessing.
After Steps 1-6 we might have a few candidates for each task. We choose the most frequent Property and Entity for CPA and
RESULTS, DISCUSSION AND CONCLUSIONS

<table>
<thead>
<tr>
<th>Round</th>
<th>CPA</th>
<th>CTA</th>
<th>CEA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1</td>
<td>P</td>
<td>Po</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2 ( &lt;1st deadline)</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2 ( &gt;1st deadline)</td>
<td>0.991</td>
<td>0.992</td>
<td>4</td>
</tr>
<tr>
<td>2 ( &gt;2nd deadline)</td>
<td>0.992</td>
<td>0.994</td>
<td>3</td>
</tr>
<tr>
<td>3 ( &lt;deadline)</td>
<td>0.949</td>
<td>0.957</td>
<td>4</td>
</tr>
<tr>
<td>3 ( &gt;deadline)</td>
<td>0.989</td>
<td>0.994</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- We used multiple APIs (without the Wikidata dump files)
- Meta-lookup resolves even difficult mistakes in a word
- Contextual matching gives F1-score above 0.97 in all tasks

F1 is F1 score
P is Precision
AF1 is approximate F1 score
AP is approximate Precision
Po is position