MantisTable V

a novel and efficient approach to Semantic Table Interpretation

Roberto Avogadro, Marco Cremaschi
Department of Computer Science, Systems and Communication (DISCo)
University of Milano - Bicocca

ISWC 2021
LamAPI
Label Matching API
LamAPI (Label Matching API) is a tool that aims to provide an easy way to store and access Wikidata and DBpedia dumps with the purpose to aid Semantic Table Interpretation tools.

The tool allows Full-text Search and Fuzzy matching over labels and provides Predicate, Entity and Concepts Matching services (using Elasticsearch).

Swagger UI of LamAPI is available at http://149.132.178.112:8097/
MantisTable V

seMantics Table
Semantic Table Interpretation: MantisTable V

MantisTable V is a automatic and unsupervised Semantic Table Interpretation approach

The approach is performed against DBpedia and Wikidata, and it can be easily adapted to any other Knowledge Graph.
MantisTable V: Key Features

- **Automatic and Unsupervised** approach
- **Conservative** approach
  - All results are sorted and scored but never removed
- **Iterative** approach
  - Tasks are not independent
  - Revision/rescoring of results
- **Row-based** approach
- High **performance** and good **accuracy**
- **Multiple KG support using LamAPI**
MantisTable V: tasks

1. Data preparation and normalisation
2. Column analysis and subject detection
3. Candidate generation
4. Cell Entity Annotation (CEA)
5. Column Predicate Annotation (CPA)
6. Column Type Annotation (CTA)
7. Revision
8. Export
Without loss of generality, let’s consider the following example:

<table>
<thead>
<tr>
<th>title</th>
<th>director</th>
<th>release year</th>
<th>domestic distributor</th>
<th>length in min</th>
<th>worldwide gross</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jurassic World</td>
<td>Colin Trevorrow</td>
<td>2015</td>
<td>Universal Pictures</td>
<td>124</td>
<td>1,670,400,637</td>
</tr>
<tr>
<td>Superman Returns</td>
<td>Bryan Singer</td>
<td>2006</td>
<td>Warner Bros.</td>
<td>154</td>
<td>391,081,192</td>
</tr>
<tr>
<td>Batman Begins</td>
<td>Christopher Nolan</td>
<td>2005</td>
<td>Warner Bros.</td>
<td>140</td>
<td>371,853,783</td>
</tr>
<tr>
<td>Avatar</td>
<td>James Cameron</td>
<td>2009</td>
<td>20 Century Fox</td>
<td>162</td>
<td>2,744,336,793</td>
</tr>
</tbody>
</table>
Bring the text to lowercase and apply to each cell the correct transformation based on datatype

The datatypes are identified using regex (e.g., boolean, date, email, geocoords, integer, float, ISBN, URL, XPath, CSS)

<table>
<thead>
<tr>
<th>title</th>
<th>director</th>
<th>release year</th>
<th>domestic distributor</th>
<th>length in min</th>
<th>worldwide gross</th>
</tr>
</thead>
<tbody>
<tr>
<td>jurassic world</td>
<td>colin trevorro</td>
<td>2015</td>
<td>universal pictures</td>
<td>124</td>
<td>1670400637</td>
</tr>
<tr>
<td>superman returns</td>
<td>bryan singer</td>
<td>2006</td>
<td>warner bros</td>
<td>154</td>
<td>391081192</td>
</tr>
<tr>
<td>batman begins</td>
<td>christopher nolan</td>
<td>2005</td>
<td>warner bros</td>
<td>140</td>
<td>371853783</td>
</tr>
<tr>
<td>avatar</td>
<td>james cameron</td>
<td>2009</td>
<td>20 century fox</td>
<td>162</td>
<td>2744336793</td>
</tr>
<tr>
<td>title</td>
<td>director</td>
<td>release year</td>
<td>domestic distributor</td>
<td>length in min</td>
<td>worldwide gross</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>--------------</td>
<td>----------------------</td>
<td>---------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>jurassic world</td>
<td>colin trevorow</td>
<td>2015</td>
<td>universal pictures</td>
<td>124</td>
<td>1670400637</td>
</tr>
<tr>
<td>superman</td>
<td>bryan singer</td>
<td>2006</td>
<td>warner bros</td>
<td>154</td>
<td>391081192</td>
</tr>
<tr>
<td>begins</td>
<td>christopher nolan</td>
<td>2005</td>
<td>warner bros</td>
<td>140</td>
<td>371853783</td>
</tr>
<tr>
<td>avatar</td>
<td>james cameron</td>
<td>2009</td>
<td>20 century fox</td>
<td>162</td>
<td>2744336793</td>
</tr>
</tbody>
</table>

1. Detection of LIT-columns uses regular expressions to identify the datatype
2. Others columns are classified as NE-columns
3. Detection of SUBJ-column, among the NE-columns, considering statistic features
For the **NE-columns**, query the KG (LamAPI) with the cells content to retrieve candidate entities and their types

A lookup query with “jurassic world” on **LamAPI Wikidata** returns the list of candidates:

- Q3512046 ("Jurassic World" - Film)
- Q21877685 ("Jurassic World" - Film)
- Q55615459 ("Jurassic World" - Amusement ride)
- ...

```json
{  
  "results": [
    {
      "id": "Q3512046",
      "name": "Jurassic World",
      "types": [
        {
          "id": "Q229390",
          "name": "3D film"
        },
        {
          "id": "Q114424",
          "name": "film"
        }
      ]
    },
    {
      "id": "Q21877685",
      "name": "Jurassic World",
      "types": [
        {
          "id": "Q11424",
          "name": "film"
        }
      ]
    },
    {
      "id": "Q55615459",
      "name": "Jurassic World",
      "types": [
        {
          "id": "Q1144661",
          "name": "amusement ride"
        }
      ]
    }
  ]
}
```
The goals of this tasks are:

1. Annotate each cell belong to a NE-column with a entity from KG
2. Collect **types** (e.g., Q11424:Film) obtained from annotations of cells
3. Collect **predicates** obtained from relations between annotations of two cells
MantisTable calls object and literals services of LamAPI to obtain a set of literal and object related to subject candidates.
Considering the winning entities from CEA phase:

<table>
<thead>
<tr>
<th>title</th>
<th>director</th>
<th>release year</th>
<th>domestic distributor</th>
<th>length in min</th>
<th>worldwide gross</th>
</tr>
</thead>
<tbody>
<tr>
<td>jurassic world</td>
<td>colin trevorrow</td>
<td>2015</td>
<td>universal pictures</td>
<td>124</td>
<td>1670400637</td>
</tr>
<tr>
<td>superman returns</td>
<td>bryan singer</td>
<td>2006</td>
<td>warner bros</td>
<td>154</td>
<td>391081192</td>
</tr>
<tr>
<td>title</td>
<td>director</td>
<td>release year</td>
<td>domestic distributor</td>
<td>length in min</td>
<td>worldwide gross</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>--------------</td>
<td>-------------------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>jurassic world</td>
<td>colin trevorw</td>
<td>2015</td>
<td>universal pictures</td>
<td>124</td>
<td>1670400637</td>
</tr>
<tr>
<td>Q3512046 (Jurassic World)</td>
<td>Q5145625 (Colin Trevorrow)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| superman returns | bryan singer     | 2006         | warner bros            | 154           | 391081192         |
| Q328695 (Superman Returns) | Q220751 (Bryan Singer) |            |                         |               |                   |

director: 2  screenwriter: 1  producer: 1
Considering the types of winning entities from the CEA phase..

```
{  "Q229390 (3D film)" : 2,
   "Q11424 (film)" : 3,
   "Q25110269 (live-action/animated film)" : 1 }
```
Consistency check the cell annotations against final results

=> recompute annotations on inconsistent annotations

e.g., a predicate of a pair of cells annotation that does not match with
the selected column predicate (CPA) may lead to a different selection
of matching entities among candidates
Export of table with annotation in different formats:

- JSON-LD
- RDF
- Turtle
tUI
table User Interface
tUI is a Web application that aims to provide a visualisation tool for STI approaches.

http://149.132.178.112:81
It works with any backend
- Data are retrieved through API endpoint
- All endpoint can be customize in a configuration file (YAML)
- Data and functionalities are based on available APIs endpoints
- All the three main tasks of the STI (CTA, CPA, CEA) are supported and annotations can be viewed directly inside the table
### Datasets

<table>
<thead>
<tr>
<th>Name</th>
<th>NTables</th>
<th>NAvgRows</th>
<th>NAvgCols</th>
<th>StdDevRows</th>
<th>StdDevCols</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>HardTablesR3_CTA_CSA_CPA_WD_Round3</td>
<td>7207</td>
<td>8.18</td>
<td>2.48</td>
<td>5.21</td>
<td>6.74</td>
<td>100.00%</td>
</tr>
<tr>
<td>Famous Movies</td>
<td>1301</td>
<td>6.18</td>
<td>2.66</td>
<td>3.41</td>
<td>6.24</td>
<td>99.98%</td>
</tr>
<tr>
<td>Popular Videogames</td>
<td>150</td>
<td>10.83</td>
<td>3.48</td>
<td>1.21</td>
<td>6.91</td>
<td>70.81%</td>
</tr>
<tr>
<td>Name</td>
<td>NCols</td>
<td>NRows</td>
<td>Status</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YE3T8MT7</td>
<td>3</td>
<td>7</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUVPK0QX</td>
<td>2</td>
<td>19</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MM8550N5</td>
<td>2</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U8BF8DFI</td>
<td>2</td>
<td>20</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DR7125C6</td>
<td>3</td>
<td>8</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WHAAXXAF</td>
<td>2</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WSWJDUNH</td>
<td>4</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SXW0B88Z</td>
<td>3</td>
<td>10</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S8C61X0A</td>
<td>2</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRTM3G3</td>
<td>3</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B8PVLJ1S</td>
<td>3</td>
<td>4</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CYY1J27G</td>
<td>3</td>
<td>5</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYRRAW3</td>
<td>3</td>
<td>14</td>
<td>✔️️️️️</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Dataset completion**
- Avg cols: 2.48
- Avg rows: 8.18
- 100.00%
<table>
<thead>
<tr>
<th>Title</th>
<th>Director</th>
<th>Year</th>
<th>Distributor</th>
<th>Length</th>
<th>Gross</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jurassic World</td>
<td>Colin Trevorrow</td>
<td>2015</td>
<td>Universal Pictures</td>
<td>124</td>
<td>1,670,400,637</td>
</tr>
<tr>
<td>Superman Returns</td>
<td>Bryan Singer</td>
<td>2006</td>
<td>Warner Bros.</td>
<td>154</td>
<td>391,081,192</td>
</tr>
<tr>
<td>Batman Begins</td>
<td>Christopher Nolan</td>
<td>2005</td>
<td>Warner Bros.</td>
<td>140</td>
<td>371,053,783</td>
</tr>
<tr>
<td>Id</td>
<td>Match</td>
<td>Name</td>
<td>Score</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------------------</td>
<td>-------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wdi:2038085</td>
<td>true</td>
<td>superman Returns</td>
<td>0.85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wdi:Q6593031</td>
<td>true</td>
<td>superman Returns</td>
<td>0.85</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Total candidates: 2
Final Remarks and Future Plans
Open research problem

- **Novel entities**
  - Identification of missing entities in the KG
  - KG evolution

- **Streaming data**
  - Row-based approach opens to incremental annotation of data streams

- **Human in the loop**
  - Revision task with domain experts to address ambiguities
Open research problem

- **Knowledge Graph**
  - Create an additional layer to unify the different KGs, and treat them as a single dataset
  - Improve the efficiency and effectiveness of candidate generation
  - Define a strategy to unify the different KGs schemas