Semantic Web 0 (2020) 1–0
IOS Press

MTab4DBpedia: Semantic Annotation for Tabular Data with DBpedia

Phuc Nguyen [a,*], Natthawut Kertkeidkachorn [b], Ryutaro Ichise [a], Hideaki Takeda [a]
[a] *National Institute of Informatics, Japan, Tokyo*
E-mails: phucnt@nii.ac.jp, ichise@nii.ac.jp, takeda@nii.ac.jp
[b] *National Institute of Advanced Industrial Science and Technology, Japan Tokyo*
E-mail: n.kertkeidkachorn@aist.go.jp

Abstract. Semantic annotation for tabular data with knowledge graphs is a process of matching table elements to knowledge graph concepts, then annotated tables could be useful for other downstream tasks such as data analytic, management, and data science applications. Nevertheless, the semantic annotations are complicated due to the lack of table metadata or description, ambiguous or noisy table headers, and table content. In this paper, we present an automatic semantic annotation system designed for the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019), called MTab4DBpedia, to generate semantic annotations for table elements with DBpedia concepts. In particular, our system could generate Cell-Entity Annotation (**CEA**), Column-Type Annotation (**CTA**), Column Relation-Property Annotation (**CPA**). MTab4DBpedia combines joint probability signals from different table elements and majority voting to solve the matching challenges on data noisiness, schema heterogeneity, and ambiguity. Results on SemTab 2019 show that our system consistently obtains the best performance for the three matching tasks: the 1st rank all rounds (the four rounds), and all tasks (the three tasks) of SemTab 2019. Additionally, this paper also provides our reflections from a participant's perspective and insightful analysis and discussion on the general benchmark for tabular data matching.

Keywords: table matching, table understanding, knowledge graph matching, DBpedia matching

1. Introduction

There are many tabular data resources published on the Web or data portals with the Open Data initiative in recent years. Those resources contain valuable information that helps establish transparency, improve human life quality, and inspire business opportunities. Although these tabular data offers enormous potential, these resources are difficult to use due to fragmentation, heterogeneous schema, missing or incomplete metadata.

One possible solution for the usability problems is to generate semantic annotation of tables, particularly matching table elements into knowledge graphs such as DBpedia. As a result, the meaning of tabular data could be interpreted or inferred by knowledge graph concepts; therefore, it is easy to use in other downstream applications such as data analytics, management, and data science applications.

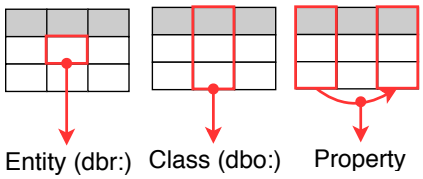
Tabular Data to Knowledge Graph Matching (SemTab 2019)[1] is a Semantic Web Challenge on matching table elements into knowledge graphs. SemTab 2019 is a systematic benchmark designed for the tabular data matching into DBpedia to promote a comparison of state-of-the-art annotation systems [1].

Fig. 1 illustrates the three tasks for SemTab 2019. Given a table data, **CEA** (Fig. 1a) is the task of assigning an entity in KG to a table cell. **CTA** (Fig. 1b) is the task of assigning a semantic type (e.g., a DBpedia class) to a table column. The relation between two table columns is assigned to a property or predicate in KG in **CPA** (Fig. 1c).

Fig. 2 depict an example of semantic annotations for tabular data. The entity of dbr:Authur_Drews is an an-

*Corresponding author. E-mail: phucnt@nii.ac.jp.

[1] SemTab 2019: http://www.cs.ox.ac.uk/isg/challenges/sem-tab/



(a) **CEA** (b) **CTA** (c) **CPA**

Fig. 1. Tabular Data Matching to DBpedia. dbr: is an entity prefix, and dbo: is an type prefix

notation for a table cell of "A. Drews". The type annotations for the column "col1" are dbo:PopulatedPlace and dbo:Place. The property of dbo:deadYear is the annotation for the relation between the column "col0" and the column "col2".

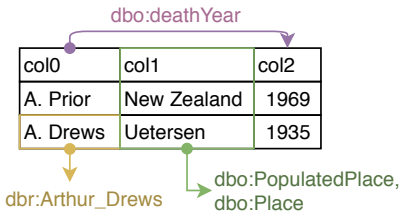


Fig. 2. Example of semantic annotation for tabular data

In this paper, we introduce MTab4DBpedia, a semantic annotation system, to address the three matching tasks of SemTab 2019. Our system combines joint probability signals from different table elements and majority voting to deal with data noisiness, schema heterogeneity, and ambiguity. Our system is inspired by the graphical probability model-based approach [2] and the signal (or confidence) propagation as in the T2K system [3]; however, our system improves matching performance with the novel solutions as follows.

- Entity Lookup: We introduce a pre-processing table pipeline, consisting of text decoding, language prediction, data type, entity type prediction and lookup aggregation, to deal with the noisiness of table data such as incorrect encoding or non-English table cells. We introduce a scoring function to estimate the uncertainty from the relevance ranking list returned from a lookup service, e.g., DBpedia, Wikipedia, or Wikidata.
- Entity Relevance Estimation: We introduce a scoring function to estimate the uncertainty from the relevance ranking list returned from a lookup service, e.g., DBpedia, Wikidata. These scores are used to aggregate entity candidates from multiple lookup services.
- Numerical Column Matching: We found that mapping cell values to corresponding values in a KG are less useful because the corresponding value in KG is rarely equal with a query value. Therefore, we adopt literal columns matching with EmbNum+ [4] to find a relevance relation (property) and aggregate these signals to enhance the overall performance.
- Column-Column-based Matching: We assume that there are logical relations between table columns; therefore, we introduce column-column-based matching for entity-entity columns, entity-literal columns and incorporate these matching results to enhance the overall matching performance.

In the end, results on SemTab 2019 show that MTab4DBpedia consistently obtains the best performance for the three matching tasks: the 1st rank all rounds (the four rounds), and all tasks (the three tasks) of SemTab 2019 [1].

MTab4DBpedia is an extended version of our work MTab [5]. This paper also provides an insightful error analysis and proposes solutions to address the task design limitations. Additionally, we also discuss our reflection on SemTab 2019.

The rest of this paper is organized as follows. In Section 2, we define the matching tasks, and describe our assumptions on MTab4DBpedia. We present the overall framework and the details of each framework's module in Section 3. In section 4, we describe the details of the matching evaluation, experimental settings, then present the results. Additionally, we also perform error analysis on SemTab 2019 dataset. We also propose some techniques to improve the current limitation of challenge design in this section. In Section 5, we discuss the related works on the task of semantic annotation for table data and summarize the participant approaches. Finally, we summarize the paper and discuss future directions, and our reflections on the challenge in Section 6.

2. Definitions and Assumptions

In this section, we provide a formal definition for the three annotation tasks of SemTab 2019 in Section 2.1. The assumptions on our system MTab4DBpedia are described in Section 2.2.

2.1. Problem Definitions

We denotes DBpedia as a knowledge graph $G = (E, T, R)$, where E, T, R are the set of entities, the set of types (or classes), and the set of relations (or predicates) respectively. e is an entity ($e \in E$), t_e is the type of the entity e ($t_e \in T$), and r is a relation of entity-entity or entity-literal ($r \in R$).

Let a table S be a two-dimensional tabular structure consisting of an ordered set of N rows, and M columns. n_i is a row of table ($i = 1...N$), m_j is a column of table ($j = 1...M$). The intersection between a row n_i and a column m_j is $c_{i,j}$ is a value of the cell $S_{i,j}$.

The tabular to KG matching problems could be formalized the three tasks as follows.

- **CEA**: matching a cell value $c_{i,j}$ into a relevance entity $e \in E$.

$$c_{i,j} \xrightarrow{\text{CEA}} E \tag{1}$$

- **CTA**: matching a column m_j into a exact relevance type and its ancestors.

$$m_j \xrightarrow{\text{CTA}} T \tag{2}$$

- **CPA**: matching the relation between two columns m_{j_1} and m_{j_2} ($j_1, j_2 \in [1, M], j_1 \neq j_2$) into a relation $r \in R$.

$$r_{m_{j_1}, m_{j_2}} \xrightarrow{\text{CPA}} R \tag{3}$$

2.2. Assumptions

We adopt the following assumptions in MTab4DBpedia.

Assumption 1. *MTab4DBpedia is built on the closed-world assumption.*

MTab4DBpedia annotate tabular data based on the knowledge graph information. Therefore, we assume that the knowledge graph (DBpedia) complete and correct. Table elements which are not available in the knowledge graph are considered as error matching samples.

Assumption 2. *The input table is represented in a vertical relational type, and the core attribute is in the first column.*

A vertical relational table contains semantic knowledge graph triples in the form of <subject, predicate, object>. Additionally, the table also has a column, which is a table core attribute. The core attribute contains entity names, and the relation between the core attribute and other table attributes (columns) represent the predicate relation between the entities (subject) and attribute values (object).

MTab4DBpedia could also be adapted to horizontal relational tables by modifying column-based attributes (vertical) to row-based attributes (horizontal).

Assumption 3. *Input tables are independent.*

We adopt this assumption since MTab4DBpedia treats input tables independently; therefore, it makes a more general matching system.

Assumption 4. *Column cell values have the same entity types and data types.*

Assumption 5. *The first row of a table (n_1) is the table header.*

Since most SemTab 2019 table headers are located in the first row, to be simple, we adopt this assumption in this paper. In practice, table headers could have more complicated structures: available or non-available, located at the beginning of the table or not, one row or multiple rows. We could add a header detection module on the prepossessing step to make MTab4DBpedia work with other dataset tables.

3. MTab4DBpedia Approach

In this section, we describe the MTab4DBpedia framework in Section 3.1. The details of each step is described in from Section 3.2 to Section 3.7.

3.1. Framework

We design our system (MTab4DBpedia) as 7-steps pipeline (Fig. 3). Step 1 is to pre-process a table data S by decoding textual data, predicting languages, data type, entity type prediction, and entity lookup. Step 2 is to estimate entity candidates for each cell. Step 3 is to estimate type candidates for columns. Step 4 is to estimate the relationship between two columns. Step 5 is to re-estimate entity candidates with confidence aggregation from step 2, step 3, and step 4. Step 6 and Step 7 are to re-estimate type and relation candidates with results from Step 5, respectively.

The following are the detailed explanations on each step of the framework.

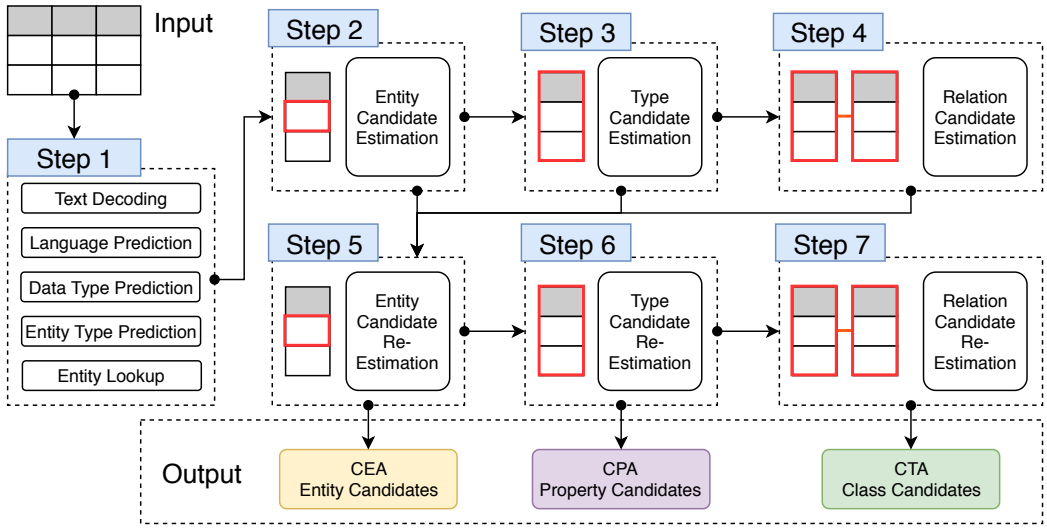


Fig. 3. MTab4DBpedia framework for tabular data matching

3.2. Step 1: Pre-processing

We perform the five processes as follows.

- **Text Decoding:** Reading table data has a problem with textual encoding, where some characters are represented as noisy sequences. Loading incorrect encoding might strongly affect the lookup performance; therefore, we used the ftfy tool [6] to fix all noisy textual data in tables.
- **Language Prediction:** We used the pre-trained fasttext models (126 MB) [7] to predict the languages for tables (concatenate all table cell values) and each cell in the table. Since table data is written in English or other languages, determining the input query's language helps the lookup tasks.
- **Data Type Prediction:** Next, we perform data type prediction to predict 13 pre-defined data types of duckling [2] for each cell value in a table $c_{i,j}$. Those types are about numerical tags, email, URL, or phone number. If there is no tag assigned, we assign this cell type as a text tag.
- **Entity Type Prediction:** For each cell value in a table $c_{i,j}$, we also perform entity type prediction with the pre-trained SpaCy models [8] (OntoNotes 5 dataset) to predict 18 entity types. If there is no tag assigned, this cell type is assigned to a text tag. We also manually map those textual entity types (11 entity types) OntoNotes 5 to some DBpedia classes.
- **Entity Lookup:** We search the relevance entity on many services including DBpedia Lookup[3], DBpedia endpoint[4]. Also, we search relevant entities on Wikipedia and Wikidata by redirected links to DBpedia to increase the possibility of finding the relevant entities. We use the language information of the table and cell values as the lookup parameters. If there is any non-English lookup URL, it is redirected to the corresponding English URL. We use α_e [5] as the limit of lookup results. The search query could be each cell value in a table $c_{i,j}$, or other neighbor cells in the same rows i.

3.3. Step 2: Entity Candidate Estimation

In this section, we explain how we estimate the entity candidates. Given a cell value $c_{i,j}$, we have a set of ranking result lists from lookup services $Q_{c_{i,j}}$. q is a list of ranking of entities ordered by degree of relevance of a lookup service, where $q \in Q_{c_{i,j}}$. In MTab4DBpedia, we adopted the four services as DBpedia lookup, DBpedia Endpoint, Wikidata lookup, and Wikipedia lookup. However, we can use any services as long as their output is a ranking list of relevance entities.

[2]Duckling link: https://github.com/facebook/duckling

[3]DBpedia Lookup, link: https://wiki.dbpedia.org/Lookup

[4]DBpedia Endpoint link: https://dbpedia.org/sparql

[5]In MTab4DBpedia, we set $\alpha_e = 100$

Denote $E_{Q_{c_{i,j}}}$ is a set of relevance entities in $Q_{c_{i,j}}$, s_e^q is a confidence score of an entity e where $e \in E_{Q_{c_{i,j}}}$. Note that the set of relevance entities is taken from lookup services ranked by these specific relevance functions. We perform a normalize step to estimate the confidence score of entity e as

$$s_e^Q = max(s_e^q) \tag{4}$$

s_e^q is the confidence score of entity e in q.

$$s_e^q = \alpha_e - rank_e \tag{5}$$

where $rank_e$ is the ranking index of entity in q, and the α_e parameter denotes for the limit of the number entity candidate in a ranking list. In our setting, we select $\alpha_e = 100$. We normalize those entity confidence score to $[0,1]$, where $Pr(E_{Q_{c_{i,j}}}|Q_{c_{i,j}}) = 1$,

$$Pr(e|Q_{c_{i,j}}) = \frac{s_e^Q}{\sum\limits_{e \in E_{Q_{c_{i,j}}}} s_e^Q} \tag{6}$$

and associate those scores as the potential probability of entities given lookup results.

3.4. Step 3: Type Candidate Estimation

Regarding assumption 4, we categorize table columns into entity columns and literal columns. We aggregate the data type (Duckling Tags and SpaCy Tags) from each cell in a column using majority voting. If the majority tag is text or entity-related, the columns are an entity column, else a numerical column. Regarding numerical columns, we perform semantic labeling with EmbNum+ method [4] to annotate relations (DBpedia properties) for numerical columns [6]. Then, we infer types (DBpedia classes) from those relations.

3.4.1. Numerical Column

The set of numerical columns in table S is M_{num}. Given a numerical column m_j, we use re-trained EmbNum+ model on DBpedia [4] to derive embedding vector for the list of all numerical values of the column and then search the corresponding relations from the database of labeled attributes [7]. The result q_{m_j} is a ranking of relevance numerical attributes in terms of distribution similarity. We use α_r as the limit for ranking results (We set $\alpha_r = 100$ in our experimental settings). The confidence score of a relation r is calculated as the following equation.

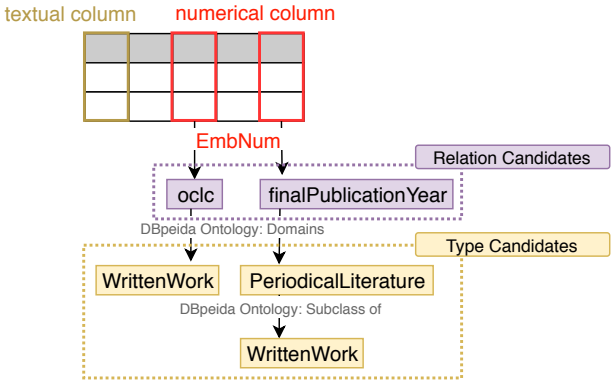


Fig. 4. Property lookup with EmbNum

$$s_r^{m_j} = \alpha_r - rank_r \tag{7}$$

where $rank_r$ is the ranking index of r. These scores are also normalized to a range of [0,1] to associate the probability of potential of relation for numerical columns $Pr(r|m_j)$.

Next, we use DBpedia endpoint to infer the classes (types) from those relations as Figure 4. $T_{q_{m_j}}$ is a set of inferred types, t denotes a type $t \in T_{q_{m_j}}$. Those types will be used for entity columns. The confidence score of types is estimated as the following equation.

$$s_{r_t} = max(s_r^{M_{num}}) \tag{8}$$

Then, we normalized those scores to [0,1] so that $Pr(T_{q_{M_{num}}}) = 1$, those confidence scores are associated as the probabilities of type potential $Pr(t|M_{num})$ given M_{num}.

3.4.2. Entity Column

Given a set of entity columns in table S is M_{ent}, we consider these signals from

1. $Pr(t|M_{num})$: the probabilities of type potential from numerical columns
2. $Pr(t|m_j, Q_{m_j})$: the probabilities of type potential aggregated from the types of entity lookup for the all cells in column m_j.

$$Pr(t|m_j, Q_{m_j}) = \sum_{c_{i,j} \in m_j} Pr(t|Q_{c_{i,j}}) \tag{9}$$

[6] We only use EmbNum+ for those columns have at least 10 numerical values

[7] We used all numerical attributes of DBpedia as the labeled data

We normalized these aggregated potentials and associates these as potential probabilities.

3. $Pr(t|m_j, SpaCy_{m_j})$: the probabilities of type potential aggregated from SpaCy entity type prediction for the all cell in column m_j. We used majority voting and normalized these voting values to [0,1]. Then, we associate those normalized voting value type potential probabilities.
4. $Pr(t|c_{1,j})$: the probabilities of type potential given header value of the column m_j. We associate the normalized Levenshtein distance as a potential probability that a type (DBpedia class) corresponds with a header value.

The probabilities of type potential are derived from the four signals as the following equation.

$$\begin{aligned} Pr(t|m_j) = w_1 Pr(t|M_{num}) + w_2 Pr(t|m_j, Q_{m_j}) \\ + w_3 Pr(t|m_j, \text{SpaCy}_{m_j}) + w_4 Pr(t|c_{1,j}) \end{aligned} \tag{10}$$

where w_1, w_2, w_3, w_4 are learnable weights. Note that some probabilities of signals might be 0 or too small, and aggregate those might add too much noise to the final aggregation. Therefore, if any signal probabilities less than β[8], we omit those signals. After aggregation, we also perform normalization for $Pr(t|m_j)$ to a range of [0,1] so that $Pr(T_{m_j}|m_j) = 1$.

3.5. Step 4: Relation Candidate Estimation

Given two columns m_{j_1} and m_{j_2}, we estimate the probabilities of relation potential of $Pr(r|m_{j_1}, m_{j_2})$. We consider two type of relation between two columns: Entity column to Entity column and Entity column to non-Entity column. To be simple, we associate the first entity column is m_{j_1}. If the second column is entity column, we denote it as $m^{ent}_{j_2}$, else $m^{non-ent}_{j_2}$.

3.5.1. Entity - Entity columns $Pr(r|m_{j_1}, m^{ent}_{j_2})$:

Given c_{i,j_1} is a cell value of the column m_{j_1} and the row r_i, c_{i,j_2} is a cell value of the column $m^{ent}_{j_2}$. We assume that there is a relation between entity candidates of c_{i,j_1} and c_{i,j_2}, therefore we use DBpedia endpoint to find how many links (relations or properties) between entity candidates of c_{i,j_1} and c_{i,j_2}. The confidence score of relation is calculated as the following equation. $s_r^{c_{i,j_1}, c_{i,j_2}} = 1$ if there is any relation between entity candidates of two columns. Then, we aggregate those scores of all rows to get the candidate score for two columns as the following equation.

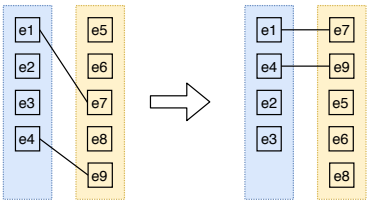


Fig. 5. Illustration of entity candidate re-ranking between two column cells

$$s_r^{m_{j_1}, m^{ent}_{j_2}} = \sum_{i \in [1,N]} s_r^{c_{i,j_1}, c_{i,j_2}} \tag{11}$$

Then, we normalize those score to a range of [0,1] so that $Pr(R_{m_{j_1}, m^{ent}_{j_2}}|m_{j_1}, m^{ent}_{j_2}) = 1$ and associate it as the probability of relation potential of Entity and Entity Columns $Pr(r|m_{j_1}, m^{ent}_{j_2})$.

Figure 5 illustrate the re-ranking between the two entity candidates from two-column cells in the same row.

3.5.2. Entity - Non-Entity columns $Pr(r|m_{j_1}, m^{non-ent}_{j_2})$:

Given c_{i,j_1} is a cell value of the column m_{j_1} and the row r_i, c_{i,j_2} is a cell value of the column $m^{non-ent}_{j_2}$. We estimate the relevance ratio between entity candidates and non-entity value c_{i,j_2}. Given an entity candidate e have pairs of relation(r_e)-values(v_e), we compare the non-entity value c_{i,j_2} with all attribute values v_e. We select those pairs have ratio larger than β. We only compare two values of c_{i,j_1} and v_e based on there data types (textual type or numerical type).

- For textual values: We use the normalized Levenshtein distance to estimate the relevance ratio between v_e and c_{i,j_2} as $s(v_e, c_{i,j_2})$.
- For numerical values: the relevance ratio is calculated as the following equation.

$$s(v_e, c_{i,j_2}) = \begin{cases} 0, \text{if } max(|c_{i,j_2}|, |v_e|) = 0 \\ \quad \text{and } |c_{i,j_2} - v_e| \neq 0 \\ 1, \text{if } max(|c_{i,j_2}|, |v_e|) = 0 \\ \quad \text{and } |c_{i,j_2} - v_e| = 0 \\ 1 - \frac{|c_{i,j_2} - v_e|}{max(|c_{i,j_2}|, |v_e|)}, \\ \quad \text{if } max(|c_{i,j_2}|, |v_e|) \neq 0 \end{cases} \tag{12}$$

[8] In MTab4DBpedia, $\beta = 0.5$

We aggregate all relevance ratio with respect to relations. Then we normalize those aggregated ratio to [0,1], and associate this as probability of relation potential. $Pr(r|m_{j_1}, m_{j_2})$. If the column of m_{j_2} is numerical columns, we also aggregate the re-calculated probability from $Pr(r|m_{j_2})$ (step 3) as the following equation.

$$Pr(r|m_{j_1}, m_{j_2}, m_{j_2}\text{is numerical}) = w_5 Pr(r|m_{j_1}, m_{j_2}) + w_6 Pr(r|m_{j_2}) \quad (13)$$

where w_5, w_6 are learnable parameters.

3.6. Step 5: Entity candidate Re-Estimation

In this step, we present a method to re-estimate the probabilities of entity candidates $Pr(e|S)$. Given a cell $S_{i,j}$ containing a cell value $c_{i,j}$ at row n_i, and column m_j, we consider these signals from:

- $Pr(e|Q_{c_{i,j}})$: The entity candidate probabilities given look up results.
- $Pr(e|m_j)$: The probabilities of entity candidates given their type's probabilities (Step 3). It could be calculated as the following equation.

$$Pr(e|m_j) = max(Pr(t_e|m_j, Q_{m_j})) \quad (14)$$

  where t_e is a type of the entity e.
- $Pr(e|c_{i,j})$: The probabilities of entity candidates given the cell value $c_{i,j}$. We get the mean ratio of the normalized Levenshtein distance, heuristic abbreviation rules (first character of words, titles, dates, time).
- $Pr(e|n_i, m_{j_1})$: The probabilities of entity candidates given cell values in a row $c_{i,j} \in n_i$. We do the same procedure as Step 4 to compare all entity values with a cell value and compute the mean probability for all cell values in a row as the following equation.

$$Pr(e|n_i, m_{j_1}) = mean(Pr(e|m_{j_1}, m_{j_2})) \quad (15)$$

  where $j_1 \neq j_2$.

Overall, the equation is as follows.

$$Pr(e|S) = w_7 Pr(e|Q_{c_{i,j}}) + w_8 Pr(e|m_j) + w_9 Pr(e|c_{i,j}) + w_{10} Pr(e|n_i, m_{j_1}) \quad (16)$$

where w_7, w_8, w_9, w_{10} are learnable parameters.

3.7. Step 6, 7: Re-Estimate Types and Relations

We select the highest probabilities of entity candidates in Step 5 for each cell $S_{i,j}$ to re-estimate types and relations with majority voting.

4. Evaluation

In this section, we first report the detail about benchmark datasets in Section 4.1, evaluation metrics in Section 4.3. The overall results are reported in Section 4.4. The detail error analysis, and improvement are described in Section 4.5.

4.1. Benchmark Datasets

The SemTab 2019 challenge contains four rounds; each round came with a different set of tables and the targets of matching for each annotation task [1]. In detail, round 1 data is extracted from the T2Dv2 dataset, round 2 is a combination of Wikipedia tables and automatically generated tables from DBpedia, round 3, and round 4 datasets also were automatically generated from DBpedia. To generate the tabular data, firstly, a list of classes and properties are gathered, then for each class, the generator selects groups of properties and using them to create "realistic" tables using SPARQL queries. Finally, these "realistic" tables were added noise into the surface textual of table cells or remove "easy" matches cells.

Table 1 reports the statistic about the SemTab 2019 dataset [1]. Round 1 dataset was extracted from the T2Dv2 dataset, a standard dataset in tabular data annotation. Round 2 dataset is the biggest and most complex one since it was combined from two different datasets, i.e., Wikipedia tables and DBpedia generated tables. Round 3 and Round 4 are generated tables, but in Round 4, the easily matched cells were removed.

Table 1
SemTab 2019 dataset

	# Table	CEA	CTA	CPA
Round 1	70	8,418	120	116
Round 2	11,925	463,773	14,561	6,762
Round 3	2,162	406,827	5,762	7,575
Round 4	818	107,352	1,732	2,747

4.2. Challenge participants

The challenge attached many research teams' attentions with seven stable systems across the four-round and annotation tasks. Table 2 depicts the number of participants in SemTab 2019.

Table 2
Number of participants in SemTab 2019

Round	1	2	3	4
CTA	13	9	8	7
CEA	11	10	8	8
CPA	5	7	7	7

4.3. Evaluation Metrics

There are four different metrics used to evaluate tabular data annotation:

F1-score is a harmonic mean of precision and recall. It is used as the primary score to measure the the performance of entity annotations (**CEA** - all rounds), relation annotations (**CPA** - all rounds), and type annotation (**CTA** - round 1). The F1 metric is calculated as follows.

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

where Precision, and Recall are calculated as follows.

$$\text{Precision} = \frac{\text{\# correct annotations}}{\text{\# annotations}} \tag{18}$$

$$\text{Recall} = \frac{\text{\# correct annotations}}{\text{\# target annotations}} \tag{19}$$

The Precision scores was also used as the secondary score in entity annotations (**CEA** - all rounds), relation annotations (**CPA** - all rounds), and type annotation (**CTA** - round 1).

Regarding the type annotation **CTA** task, there are two metrics designed to measure the hierarchical of class annotations (Average Hierarchical - AH) and perfect class annotations (Average Perfect - AP). The AH score is used as the primary score, while the AP score is used as the secondary score for round 2, 3, 4 of **CTA** task.

Denote that the list of target columns is T. A column annotation is denoted as a, and the number of perfect annotations denotes as p_a, the number of OK annotation denotes as o_a, and the number of the wrong annotation denote as w_a. The equations of the AH score and AP score are described as follows. The full results

$$\text{AH} = \frac{\sum_{a \in T} p_a + 0.5 * o_a - w_a}{|T|} \tag{20}$$

$$\text{AP} = \frac{\sum_{a \in T} p_a}{\sum_{a \in T} p_a + o_a + w_a} \tag{21}$$

4.4. Experimental Results

Table 3 (CEA), Table 4 (CTA), Table 5 (CPA) reports the overall results of MTab4DBpedia for three matching tasks in the four rounds of SemTab 2019. These results show that MTab4DBpedia achieves the best performance performances on the three matching tasks and the four rounds of the challenges. The full results are reported in the challenge websites[9].

The MTab4DBpedia performance might be explained in part by tackling the major problems on entity lookup and aggregation. We performed language prediction and lookup with the language parameter. Moreover, our system is built on top of multiple lookup services such as Wikipedia, Wikidata, DBpedia; therefore, it increases the possibility of finding the relevant entities. Additionally, we adopted many new signals from table elements such as literal matching, column-column matching.

4.5. Error Analysis and Improvement

In this section, we analyze the error cases of entity matching (Section 4.5.1), type matching (Section 4.5.2), and relation matching (Section 4.5.3) of MTab4DBpedia system. We report the analysis results on Round 2, 3, 4 dataset due to the larger number of tables and variety table size [1]. The Round 1 dataset is a subset of a T2Dv2 dataset; this dataset is used as a demo sample data about the annotation tasks.

[9]Results: http://www.cs.ox.ac.uk/isg/challenges/sem-tab/2019/results.html

Table 3
CEA results in F1 score (the primary metric) and Precision for the four rounds of SemTab 2019 (7 stable systems)

	F1 score				Precision			
Round	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
MTab	**1.000**	**0.911**	**0.97**	**0.983**	**1.00**	**0.911**	**0.97**	**0.983**
CSV2KG	0.448	0.883	0.962	0.907	0.627	0.893	0.964	0.912
Tabularisi	0.884	0.826	0.857	0.803	0.908	0.852	0.866	0.813
MantisTable	1.000	0.614	0.633	0.973	1.00	0.673	0.679	0.983
LOD4ALL	0.852	0.757	0.828	0.648	0.874	0.767	0.833	0.654
ADOG	0.657	0.742	0.912	0.835	0.673	0.745	0.913	0.838
DAGOBAH	0.897	0.713	0.725	0.578	0.941	0.816	0.745	0.599

Table 4
CTA results in F1 and Precision for the round 1 and AH score, and AP score for the rounds 2, 3, 4 of SemTab 2019 (7 stable systems)

	AH score				AP score			
Round	**1(F1)**	**2**	**3**	**4**	**1(Precision)**	**2**	**3**	**4**
MTab	**1.000**	**1.414**	**1.956**	**2.012**	**1.000**	**0.276**	**0.261**	**0.300**
CSV2KG	0.833	1.376	1.864	1.846	0.833	0.257	0.247	0.274
Tabularisi	0.825	1.099	1.702	1.716	0.825	0.261	0.277	0.325
MantisTable	0.929	1.049	1.648	1.682	0.933	0.247	0.269	0.322
LOD4ALL	0.850	0.893	1.442	1.071	0.850	0.234	0.260	0.386
ADOG	0.829	0.713	1.409	1.538	0.851	0.208	0.238	0.296
DAGOBAH	0.644	0.641	0.745	0.684	0.580	0.247	0.161	0.206

Table 5
CPA results in F1 score (primary) and Precision for the four rounds of SemTab 2019 (7 stable systems)

	F1 score				Precision			
Round	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
MTab	**0.987**	**0.881**	**0.844**	**0.832**	**0.975**	**0.929**	**0.845**	**0.832**
CSV2KG	-	0.877	0.841	0.830	-	0.926	0.843	0.835
Tabularisi	0.606	0.79	0.827	0.823	0.638	0.792	0.83	0.825
MantisTable	0.965	0.46	0.518	0.787	0.991	0.544	0.595	0.841
LOD4ALL	-	0.555	0.545	0.439	-	0.941	0.853	0.904
ADOG	-	0.459	0.558	0.75	-	0.708	0.763	0.767
DAGOBAH	0.415	0.533	0.519	0.398	0.347	0.919	0.826	0.874

4.5.1. CEA: Entity Matching

Regarding entity matching ground truth, we found that many samples are inconsistent URI encoded and decoded representation. For example, an entity URI of dbr:Angélica_Rivera could be encoded as "dbr:Ang%C3%A9lica_Rivera" and decoded as "dbr:Angélica_Rivera". The ground truth of CEA contains a mixture between encoded URI and decoded URI. According to URI encoding of DBpedia, the encoding URI (percent-encoding) is not encouraged [10]. We verify how many samples do not have decoded URI in CEA ground truth in Table 6. Note that we only focus on the large dataset, such as Round 2, 3, 4.

Table 6
Number of none decoded URI samples in CEA ground truth in SemTab 2019

Ground Truth	# Samples	# No decode samples
Round 2	463,773	12,200 (2.63%)
Round 3	406,827	3,273 (0.8%)
Round 4	107,352	174 (0.16%)

To make the URI consistency, we provide the encoded and decoded URI for each URI in the original CEA ground truth. Then, we measure the

[10] DBpedia URI encoding: https://wiki.dbpedia.org/uri-encoding

MTab4DBpedia performance on the new ground truth called EDCEA_GT (Encoded and Decoded CEA ground truth). In Table 7, the performance of MTab4DBpedia slightly improve when testing on EDCEA_GT.

Table 7
Comparison of MTab4DBpedia performance in the original CEA ground truth (CEA_GT) and the new CEA ground truth (EDCEA_GT)

	CEA_GT	EDCEA_GT
Round 2	0.911	0.916
Round 3	0.970	0.978
Round 4	0.983	0.984

4.5.2. CTA: Type Matching

This section reports two types of CTA errors of the MTab4DBpedia system: incorrect type annotations and incorrect branch annotations. The incorrect type annotations are those annotations assigned in different types. For example, the annotation is [dbo:TelevisionShow, dbo:Work] where the ground truth is [dbo:Person]. The incorrect branch annotations are those annotations assigned in the same abstract type, but different subtypes such as the annotation are [dbo:Writer, dbo:Person] where the ground truth is [dbo:Person]. In this case, the annotation of dbo:Writer is incorrect, but the abstract type dbo:Person is correct.

Table 8
Error analysis of MTab4DBpedia on CTA tasks

Errors	Type	Branch	Total
Round 2	697 (40.96%)	1005 (59.04%)	1702
Round 3	72 (14.94%)	410 (85.06%)	482
Round 4	20 (35.71%)	36 (64.29%)	56

Table 8 reports the details statistic of total errors and two error types of MTab4DBpedia for CTA tasks. The incorrect branch annotation has a larger number of errors than the type annotations. The number of error annotations in Round 4 is the smallest one. As the majority voting on entity types rely on the CEA entity annotation tasks, the CEA results also got the highest performance in the Round 4 dataset. Improving the entity annotations (CEA) could improve the performance of type annotations (CTA).

4.5.3. CPA: Relation Matching

Regarding relation matching, we found that the ground truth annotations do not have the equivalent relations. For example, the relation of dbo:team has its equivalent relation as dbo:club. The direct equivalent properties in DBpedia endpoint are (dbo:team, dbo:club), (dbo:language, dbo:deFactoLanguage, dbo:jureLanguage), and (dbo:area, dbo:landArea, dbo:waterArea). We add those direct equivalent properties into the ground truth (CPA_GT) and associate the new ground truth as DECPA_GT (Direct Equivalent CPA Ground Truth).

Table 9
Comparison between MTab4DBpedia performance in the original CPA_GT and the new ground truth DECPA_GT in the Round 2, 3, and 4 data

	# Errors		F1 score	
	CPA_GT	DECPA_GT	CPA_GT	DECPA_GT
2	1,090	388	0.839	0.888
3	1,208	939	0.844	0.875
4	457	301	0.833	0.890

Table 9 reports a comparison between MTab4DBpedia performance in the original CPA_GT and the new ground truth DECPA_GT. The performance of MTab4DBpedia on CPA significantly improve when tested in DECPA_GT.

Due to the incompleteness of DBpedia, we found that there are other indirect equivalent relations in DBpedia. For example, dbo:deathCause and dbo:causeOfDeath have the same equivalent property of wikidata:P509 (cause of death). The problem of knowledge graph completion is not the main focus of this work, but we can expect the improvement of relation annotations when the completeness of DBpedia is improved.

5. Related Work

In this section, we reviewed the other systems participating in SemTab 2019. Also, we discussed related works on the tabular data annotation task.

5.1. SemTab 2019 systems

This section describes the six other systems frequently participants for all rounds of SemTab 2019 challenges.

In general, all of the participants start to generate entity candidates by lookup table cell values or searching those values in the local index with Elastic Search in DBpedia, Wikidata. The details about the lookup services are reported in Table 10. Then, the type candi-

Table 10
Comparison of entity candidate generation methods of SemTab 2019 participants

	MTab4DBpedia	CSV2KG	TabularISI	MantisTable	LOD4ALL	ADOG	DAGOBAH
URI heuristic*	x	✓	x	x	✓	x	x
DBpedia SPARQL	✓	x	x	✓	x	x	✓
DBpedia Lookup	✓	✓	x	x	x	x	x
Dbpeida Spotlight	x	✓	x	x	x	x	x
Wikidata SPARQL	✓	x	✓	x	x	x	✓
Wikipedia (CirrusSearch)	x	x	x	x	x	x	✓
Wikipedia (Multilingual)	✓	x	x	x	x	x	x
DBpedia Elastic Search	x	x	✓	x	x	✓	x
Wikidata Elastic Search	x	x	✓	x	x	x	✓
LOD4ALL Elastic Search	x	x	x	x	✓	x	x

dates and relation candidates are estimated using the entity candidates. Then, re-estimate the entity candidates with the type and relation candidates.

CSV2KG (IDLAB) first search on DBpedia lookup and DBpedia Spotlight to generate entity candidates [9]. The type candidates and relation annotations are estimated using majority voting approaches based on entity candidates. Then, the entity annotations are estimated using the information of relation candidates. Finally, type annotations are estimated using entity annotations.

Tabular ISI approach first generates entity candidates with Wikidata API and Elastic Search on entity labels of Wikidata, DBpedia. Second, the authors use the heuristics TF-IDF approach and machine learning (neural network ranking) model to select the best candidate for the entity annotation task [10]. The type annotations are estimated with the results from entity annotations with hierarchy searching on common classes. The relation annotations are estimated by finding the relation between entity candidates of the primary and secondary columns or value matching between the primary and secondary columns' value.

Mantis Table performs column analysis, including predicting name entity columns, literal columns, and subject column, then mapping between columns into concepts in DBpedia [11][11]. The relationships between the core attribute column and other columns are estimated based on predicate context and predicate frequency of column value and candidate predicates. Finally, entity linking is performed using the results from previous steps for cell value disambiguation. The relation annotations are estimated by getting the maximum frequency of relation candidates in the entity linking phase. To estimate type annotations, the authors calculate the hierarchical path score of entity types from entity annotations. Then type annotations are estimated on the maximum of the path score.

DAGOBAH performs entity linking with a lookup on Wikidata and DBpedia as well as voting mechanisms [12]. The authors used Wikidata entity embedding to estimate the entity type candidates, assuming that the same column's entities should be closed in the embedding spaces as they share semantic meanings.

LOD4ALL uses a combination of direct search (SPARQL ASK on dbr:"query"), keyword search (Abbreviation of Human name), and Elastic Search to find entity candidates [13]. The entity candidates will be used to estimate type annotations with majority voting. Then, the system determines the entity annotations with the type constraints. Finally, the relation annotations are estimated by a column-based majority voting with entity annotations of each table row.

ADOG focuses on entity annotation with Elastic Search on an integrated ontology (DBpedia sub-graph) using a NoSQL database named ArangoDB [14]. The system estimates entity annotation using Levenshtein distance, and the results of type and relation annotations are estimated from entity annotations.

In summary, all these methods focus on lookup services of DBpedia, Wikidata, but such services do not usually return relevance entities for non-English queries. In MTab4DBpedia, we address non-results entities' problems at the lookup step by predicting the language used in a table and lookup on multiple services using language parameters. The aggregation of these lookup results increases the possibility of finding relevance entities for general tabular data.

Moreover, these tabular data contain many numerical attributes that help us use semantic labeling results for numerical attributes. In MTab4DBpedia, we

[11] Web interface: http://zoo.disco.unimib.it/mantistable/

aggregate signal from the result of semantic labeling for numerical attributes (columns) using EmbNum+ [4] (deep metric for distribution similarity calculation). Additionally, we also extract the column-column relation signal to enhance overall matching performance.

5.2. Other Tabular Data Annotation Tasks

The tabular data annotation tasks could be categorized as structure or semantic annotation.

The structure annotation contains many tasks as table type prediction [15], data type prediction, table header annotation, core attribute prediction, and holistic matching across tables [16]. In the SemTab 2019, most tables are represented as a vertical relation type; headers are located at the first row of tables, and the core attribute is in the first table column. Therefore, in MTab4Wikidata, we make assumptions on the table header (Assumption 5), and the core attribute (Assumption 2).

Previous studies on table semantic annotation involves of schema-level matching e.g, tables to classes [3], columns to properties [3, 4, 17, 18] or classes [19], and data-level matching e.g., rows [3, 20] or cells [2, 19] to entities. SemTab 2019 also has schema annotation as the CTA task, data annotation as the CEA task, and a novel CPA task as column relation annotation.

6. Conclusion

In this paper, we present MTab4DBpedia for the SemTab 2019 challenge on tabular data to DBpedia matching. MTab4DBpedia address the three annotations tasks with the novel solutions on entity lookup, entity relevance estimation, numerical column matching, and column-column-base matching. The result of SemTab 2019 shows that our system consistently achieves the best performance of the three matching tasks on four rounds of the challenge.

6.1. Limitations

Since MTab4DBpedia is built on top of lookup services, the upper bound of accuracy strongly relies on the lookup results. In MTab4DBpedia, it is computation-intensive because of aggregating the confidence signals from many parts of the table. Therefore, MTab4DBpedia is not suitable for the real-time application, where we need to get the result as fast as possible. MTab4DBpedia could be modified to match only some parts of the table to reduce the processing time as Table Miner+ [19]. However, we find that this is a trade-off between effectiveness and efficiency when using Table Miner+ [19] method. A concrete analysis of the trade-off issue is left as our future investigation.

6.2. Future Works

MTab4DBpedia could be improved in many dimensions, such as effectiveness, efficiency, and generality. Regarding efficiency, MTab4DBpedia could be modified in as parallel processing fashion since the lookup steps, and these probability estimations in Step 2, 3, and 4 are independence. Regarding effectiveness, MTab4DBpedia performance could be improved by relaxing our assumptions:

- Assumption 1: The closed-world assumption might not hold in practice. Improving the completeness and correctness of knowledge graphs might improve MTab4DBpedia performance.
- Assumption 2: Classifying table types before matching could help to improve MTab4DBpedia performance.
- Assumption 3: In reality, some tables could have shared schema. For example, tables on the Web could be divided into many web pages; therefore, we can expect improving matching performance by stitching those tables on the same web page (or domain) [16], [21]. Therefore, performing holistic matching could help improve MTab4DBpedia performance.
- Assumption 5: Correctly recognize table headers could help to improve MTab4DBpedia performance.

6.3. Challenge reflections

This session discusses our reflections on SemTab 2019 challenge

Benchmarking Value From our perspective, SemTab 2019 plays a vital role in benchmarking tabular data annotation tasks. Due to the differences in benchmark settings, tabular datasets, and target matching knowledge bases in the literature, there is a need for a general benchmark for tabular data matching tasks to promote a fair comparison of annotation systems. This challenge's results reflect the practical results on many matching techniques, or the importance of features do matter for the tabular matching.

DBpedia as the target knowledge graph The choice of DBpedia as the target matching reflects the low update knowledge graph (DBpedia update after 1-2 years). In real-world practice, many knowledge graphs change rapidly, such as Wikidata. We will have a different set of challenges in matching these fast-evolving knowledge graphs.

Acknowledgements

We would like to thank the SemTab 2019 challenge organizers for their organizing the successful challenge. We also thank IBM Research and SIRIUS for their sponsorship for the challenge.

References

[1] O. Hassanzadeh, V. Efthymiou, J. Chen, E. Jiménez-Ruiz and K. Srinivas, SemTab2019: Semantic Web Challenge on Tabular Data to Knowledge Graph Matching - 2019 Data Sets, Zenodo, 2019. doi:10.5281/zenodo.3518539.

[2] G. Limaye, S. Sarawagi and S. Chakrabarti, Annotating and Searching Web Tables Using Entities, Types and Relationships, *PVLDB* **3**(1) (2010), 1338–1347. doi:10.14778/1920841.1921005. http://www.vldb.org/pvldb/vldb2010/pvldb_vol3/R118.pdf.

[3] D. Ritze, O. Lehmberg and C. Bizer, Matching HTML Tables to DBpedia, in: *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS 2015, Larnaca, Cyprus, July 13-15, 2015*, 2015, pp. 10:1–10:6. doi:10.1145/2797115.2797118.

[4] P. Nguyen, K. Nguyen, R. Ichise and H. Takeda, EmbNum+: Effective, Efficient, and Robust Semantic Labeling for Numerical Values, *New Generation Computing* (2019). doi:10.1007/s00354-019-00076-w.

[5] P. Nguyen, N. Kertkeidkachorn, R. Ichise and H. Takeda, MTab: Matching Tabular Data to Knowledge Graph using Probability Models, *CoRR* **abs/1910.00246** (2019). http://arxiv.org/abs/1910.00246.

[6] R. Speer, ftfy, 2019, Version 5.5. https://github.com/LuminosoInsight/python-ftfy.

[7] E. Grave, T. Mikolov, A. Joulin and P. Bojanowski, Bag of Tricks for Efficient Text Classification, in: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, 2017, pp. 427–431. https://www.aclweb.org/anthology/E17-2068/.

[8] M. Honnibal and I. Montani, spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, 2017, To appear. https://spacy.io/.

[9] G. Vandewiele, B. Steenwinckel, F.D. Turck and F. Ongenae, CVS2KG: Transforming Tabular Data into Semantic Knowledge, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/IDLab.pdf.

[10] A. Thawani, M. Hu, E. Hu, H. Zafar, N.T. Divvala, A. Singh, E. Qasemi, P. Szekely and J. Pujara, Entity Linking to Knowledge Graphs to Infer Column Types and Properties, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/Tabularisi.pdf.

[11] M. Cremaschi, R. Avogadro and D. Chieregato, MantisTable: an Automatic Approach for the Semantic Table Interpretation, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/MantisTable.pdf.

[12] Y. Chabot, T. Labbe, J. Liu and R. Troncy, DAGOBAH: An End-to-End Context-Free Tabular Data Semantic Annotation System, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/DAGOBAH.pdf.

[13] H. Morikawa, F. Nishino and N. Igata, Semantic Table Interpretation using LOD4ALL, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/LOD4ALL.pdf.

[14] D. Oliveira and M. d'Aquin, ADOG - Anotating Data with Ontologies and Graphs, 2019, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, ISWC. http://www.cs.ox.ac.uk/isg/challenges/sem-tab/papers/ADOG.pdf.

[15] K. Nishida, K. Sadamitsu, R. Higashinaka and Y. Matsuo, Understanding the Semantic Structures of Tables with a Hybrid Deep Neural Network Architecture, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, 2017, pp. 168–174. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14396.

[16] O. Lehmberg and C. Bizer, Stitching Web Tables for Improving Matching Quality, *PVLDB* **10**(11) (2017), 1502–1513. doi:10.14778/3137628.3137657. http://www.vldb.org/pvldb/vol10/p1502-lehmberg.pdf.

[17] M. Pham, S. Alse, C.A. Knoblock and P.A. Szekely, Semantic Labeling: A Domain-Independent Approach, in: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, 2016, pp. 446–462. doi:10.1007/978-3-319-46523-4_27.

[18] J. Chen, E. Jiménez-Ruiz, I. Horrocks and C.A. Sutton, ColNet: Embedding the Semantics of Web Tables for Column Type Prediction, in: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, January 27 - February 1, 2019, Honolulu, Hawaii, USA*, 2019, pp. 29–36. doi:10.1609/aaai.v33i01.330129.

[19] Z. Zhang, Effective and efficient Semantic Table Interpretation using TableMiner$^+$, *Semantic Web* **8**(6) (2017), 921–957. doi:10.3233/SW-160242.

[20] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro and V. Christophides, Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings, in: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, 2017, pp. 260–277. doi:10.1007/978-3-319-68288-4_16.

[21] D. Ritze, Web-Scale Web Table to Knowledge Base Matching, PhD thesis, University of Mannheim, Germany, 2017. https://ub-madoc.bib.uni-mannheim.de/43123.