KGCODE-Tab Results for SemTab 2022

Xinhe Li[1], Shuxin Wang[1], Wei Zhou[3], Gongrui Zhang[2], Chenghuan Jiang[2], Tianyu Hong[1] and Peng Wang[1,2,3,*]

[1]School of Computer Science and Engineering, Southeast University, China
[3]College of Software Engineering, Southeast University, China
[2]Chien-Shiung Wu College, Southeast University, China

Abstract

This paper presents the results of KGCODE-Tab in the tabular data to knowledge graph matching contest SemTab 2022. As an efficient tabular data linking system, KGCODE-Tab is intended to participate in three tasks of the content: Column Type Annotation (CTA), Cell Entity Annotation (CEA), and Columns Property Annotation (CPA). The specific techniques used by KGCODE-Tab will be introduced briefly. The strengths and weaknesses of KGCODE-Tab will also be discussed.

Keywords

Tabular Data, Knowledge Graph, Entity Linking, KGCODE-Tab, Semantic Annotation

1. Presentation of the system

KGCODE-Tab, as a novel table annotation system, can efficiently deal with three tabular data to knowledge graph matching (TDKGM) [1] tasks: Column Type Annotation (CTA), Cell Entity Annotation (CEA), and Columns Property Annotation (CPA). Our system fully utilizes the structure of tabular data and the information provided by knowledge graphs (KGs). Experimental results on the SemTab 2022[1] datasets demonstrate that KGCODE-Tab has excellent disambiguation ability and achieves outstanding performance with less query time.

1.1. State, purpose, general statement

The core principle of matching strategies of KGCODE-Tab is utilizing the structure of tabular data and the information provided by KGs correctly and effectively. KGCODE-Tab mainly consists of three modules: tabular data preprocessing, entity disambiguation, and task analysis.

SemTab@ISWC 2022, October 23–27, 2022, Hangzhou, China (Virtual)
*Corresponding author.
lixinhe669@gmail.com (X. Li); shuxinwang662@gmail.com (S. Wang); zhouweiseu@seu.edu.cn (W. Zhou); grzhang@seu.edu.cn (G. Zhang); quadnucyard@gmail.com (C. Jiang); tianyuhong677@gmail.com (T. Hong); pwang@seu.edu.cn (P. Wang)
https://github.com/Xinhe-Li (X. Li); https://github.com/A-BigTree (S. Wang); https://github.com/MyWhiteLip (W. Zhou); https://github.com/TideDra (G. Zhang); https://github.com/QuadnucYard (C. Jiang); https://github.com/Tianyu-Hong (T. Hong)
0000-0002-6299-4229 (X. Li); 0000-0002-3677-8477 (S. Wang); 0000-0002-4558-245X (W. Zhou); 0000-0002-8342-5834 (G. Zhang); 0000-0002-3583-3569 (C. Jiang); 0000-0002-3773-7108 (T. Hong)

[1]Semantic Web Challenge on Tabular Data to Knowledge Graph Matching-https://www.cs.ox.ac.uk/isg/challenges/sem-tab/

KGCODE-Tab combines several effective tabular data preprocessing techniques, which are fundamental for TDKGM. We analyze the structure of tabular data, which is helpful to extract the subject column and non-subject columns, correct the spelling of texts in cells, and recall all candidate entities and their information needed in the later modules. In the entity disambiguation module, preliminary scores are assigned to all candidate entities of the cells in the subject column, based on the similarities between tabular cells and property values in KGs. In each task, a ranking algorithm is designed according to the preliminary scores, and finally we obtain the semantic annotation based on the ranks. KGCODE-Tab separates the look-up step and entity linking step, the latter can directly use the intermediate results produced by the former in JSON files.

In SemTab 2022, KGCODE-Tab is an efficient tabular data linking system, and some algorithms and matching strategies of it have been designed for high efficiency.

1.2. Specific techniques used

KGCODE-Tab aims to provide high-quality semantic annotation of tabular data. The main specific techniques used by KGCODE-Tab are as follows.

1.2.1. Table Structure Analysis

Firstly, KGCODE-Tab classifies each column into entity column and non-entity column. It employs *spaCy*[2], a python package for Named Entity Recognition (NER), to give each cell a tag. A cell is an entity cell if it is tagged with `PERSON`, `NORP`, `FAC`, `ORG`, `GPE`, `LOC`, `PRODUCT`, `EVENT`, `WORK_OF_ART`, `LAW` or `LANGUAGE`. A cell is a non-entity cell if it is tagged with `DATE`, `TIME`, `PERCENT`, `MONEY`, `QUANTITY`, `ORDINAL` or `CARDINAL`. Cells that cannot be recognized by *spaCy* are classified into entity cells to prevent omissions. Then a column is an entity column if more than half of its cells (except the header) are entity cells. Otherwise, it is a non-entity column.

Secondly, KGCODE-Tab selects the subject column from the entity columns. It defines the *Column Entropy*, which describes the diversity of contents in a column. The subject column commonly has a higher value of the *Column Entropy*. If more than one subject columns exist, then KGCODE-Tab selects the one with the smallest index.

1.2.2. Spell Correction

Tables on the Internet usually have misspelled words, and researches [2, 3] show that spelling mistakes can make a huge difference to entity recall. Some systems [4, 5] remove special characters in the text, but have no idea about the wrong words. Inspired by [6], KGCODE-Tab utilizes search engines to find the correct words.

For a tabular cell c_{ij}, KGCODE-Tab uses Bing[3] to search it and obtains the result page in HTML format. Secondly, it extracts the titles of websites in the HTML and splits them into words $\mathscr{W} = \{w_1, w_2, \ldots, w_n\}$, where n is the total number of words. Thirdly, it calculates the *Levenshtein Distance* between $w_i, i = 1, 2 \ldots, n$ and c_{ij}. Finally, the word with the shortest *Levenshtein Distance*

[2] https://github.com/explosion/spaCy
[3] https://www.bing.com/search

to c_{ij} is selected as the correct mention of c_{ij}, and words whose *Levenshtein Distance* to the correct word are no more than 2 are also appended to the list of candidate mentions of c_{ij}, preventing omissions.

1.2.3. Entity Recall

Entity recall aims to select several candidate entities from a given KG. If the system cannot even recall the ground truth entities, then all the subsequent work is in vain. For the data source of KG, Some systems [7, 8, 9] build their database using the Wikidata local dump. However, the method requires high storage and IO performance of computers due to the huge size of local dump files. Therefore, we use the look-up services *MediaWiki Action API*[4] and *DBpedia Lookup*[5] to access the data of KGs online. We use 100 threads in entity query to improve query speed and obtain up to 50 candidate entities for each query text.

Furthermore, we find that the look-up services of KGs (Wikidata/DBpedia) are sensitive to the noise in the query text, such as adverbs, adjectives, prepositions, and so on. They may lead to wrong or empty results.

To tackle this problem, we introduce the tokenization technique. For the text of cell c_{ij} with l words $\mathbf{t} = [t_1, t_2, \dots, t_l]$, KGCODE-Tab constructs a query set $\mathcal{Q} = \{\mathbf{q}_{i:j} = [t_i, t_{i+1}, \dots, t_j] \mid i, j = 1, 2, \dots, l \text{ and } i \leqslant j\}$. Then it sends each $\mathbf{q}_{i:j}$ in $\mathcal{Q}$ to the spell correction module and obtains the candidate mention set $\mathcal{M}$ of c_{ij}. Finally, it sends $\mathcal{M}$ into the KGs API and gets the candidate entities set $\mathcal{E}$. It also collects the information of each entity into a dictionary containing its label, description, statements, identifiers, and so on.

1.2.4. Entity Disambiguation

Entity disambiguation is to select the ground truth entity from candidate entities. The architecture of existing systems can be classified into two categories: Graph-based [7, 8, 10] and Score-base [2, 4, 5, 11], and we design an algorithm to calculate the similarity score.

Commonly, a table has at least one subject column, and the others are non-subject columns. The non-subject columns are generally properties of subject columns. Therefore, KGCODE-Tab can exclude some candidate entities of subject columns by comparing their properties with the content of related non-subject columns. There are mainly six data types in Wikidata: *wikibase-entityid*, *string*, *time*, *globecoordinate*, *quantity*, and *multilingualtext*, so we need to design different formulas to calculate the similarity score according to different data types. Let an entity e has P properties, and v_k denotes the k-th property.

For the string and multilingualtext data types, it is enough to rely on *Levenshtein Distance*. For the wikibase-entityid data type, they need to be converted to labels firstly. The similarity score formula is shown as follows:

$$Sim(c_{ij}, v_k) = \begin{cases} LevRatio(c_{ij}, v_k), & LevRatio(c_{ij}, v_k) \geqslant \alpha \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where the optimal value of parameter α is 0.98 which is obtained by experiments.

[4] https://wikidata.org/w/api.php
[5] https://lookup.dbpedia.org/

For the quantity data type, we define the *Number Relevance Degree* (NRD) which is shown as follows:

$$NRD(a,b) = \begin{cases} 1 - \frac{|a-b|}{\max(|a|,|b|)}, & ab \neq 0 \text{ and } 1 - \frac{|a-b|}{\max(|a|,|b|)} \geqslant \beta \\ 1 - |a-b|, & ab = 0 \text{ and } 1 - |a-b| \geqslant \beta \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$Sim(c_{ij}, v_k) = NRD(c_{ij}, v_k) \tag{3}$$

where the optimal value of parameter β is 0.98 which is also obtained by experiments.

For the globecoordina data type which contains longitude and latitude, we directly use NRD to calculate the similarity score. The similarity score formula is shown as follows:

$$Sim(c_{ij}, v_k) = \max\left(NRD(c_{ij}, v_k^{la}), NRD(c_{ij}, v_k^{lo})\right) \tag{4}$$

For the time data type, we define a list $\mathbf{T}$ which contains year, month, day, hour, minute, and second to represent the time value. In tabular data, we use regular expressions for extracting time information as a $\mathbf{T}$. The similarity score formula is shown as follows:

$$Sim(c_{ij}, v_k) = \begin{cases} 1, & \mathbf{T}_{c_{ij}} \subseteq \mathbf{T}_{v_k} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

After the similarity scores calculation, each candidate entity has a final score calculated by the formula:

$$FS(e) = \frac{1}{N-1} \sum_{j=1, j \neq s}^{N} \max_{v_k \in P_e} Sim(c_{ij}, v_k) \tag{6}$$

where e is the candidate entity of the i-th cell in the subject column, s denotes the column index of subject column, and P_e is the set of properties in e.

1.2.5. Task Analysis

In our system, we utilize a cooperative score mechanism. Let $M(e_i^k, c_{ij})$ and $M(e_i^k, e_{ij}^{k'})$ denote the matching score of (e_i^k, c_{ij}) or $(e_i^k, e_{ij}^{k'})$ used later. We use a normalization function

$$\phi(x) = (ax)^b \tag{7}$$

to widen the gap between high and low matching score, where $a = 1.1$ and $b = 8$.

Column Type Annotation Let e_i^k denote the k-th candidate entity of the i-th cell in the subject column. Then the set of candidate types is $\mathscr{C}_{\text{sub}} = \{t | (e_i^k, \mathit{InstanceOf}, t) \in KG, i = 1, 2, \ldots, m, k = 1, 2, \ldots N(c_i)\}$, where $N(c_i)$ is the number of candidate entities of the i-th cell. We assign a score to each type t in $\mathscr{C}_{\text{sub}}$ by Eq.9.

$$I_{\text{sub}}(t, e_i^k) = \begin{cases} \frac{1}{N-1} \sum_{j \neq s} M(e_i^k, c_{ij}), & (e_i^k, \mathit{InstanceOf}, t) \in KG \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$CTAScore_{sub}(t) = \sum_{i=1}^{m} \max_{k=1}^{N(c_i)} \phi\left(I_{\text{sub}}(t, e_i^k)\right) \tag{9}$$

For non-subject columns, the score of candidate types in $\mathscr{C}_{\text{non}}$ are assigned by Eq.11.

$$I_{\text{non}}(t, e_i^k, e_{ij}^{k'}) = \begin{cases} M(e_i^k, e_{ij}^{k'}), & (e_{ij}^{k'}, InstanceOf, t) \in KG \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

$$CTAScore_{non}(t_j) = \sum_{i=1}^{M} \max_{k,k'} \phi\left(I_{\text{non}}(t_j, e_i^k, e_{ij}^{k'})\right) \tag{11}$$

Cell Entity Annotation For an entity in the subject column, we enumerate all types t^u of candidates to take advantage of CTA scores, as shown in Eq.12, where the parameter λ is a cooperative factor set to 0.1. We skip the items that makes $I_{\text{sub}}(\cdot,\cdot)$ or $M(\cdot,\cdot)$ equals 0.

$$CEAScore_{sub}(e_i^k) = \max_{t,u} \left\{\phi\left(I_{\text{sub}}(t^u, e_i^k)\right) + \lambda \cdot CTAScore_{sub}(t^u)\right\} \tag{12}$$

For a non-subject column with index j, we give the entity e_{ij}^k score by Eq.13.

$$CEAScore_{non}(e_{ij}^{k'}) = \max_{k'=1}^{N(c_{ij})} \left\{\phi\left(M(e_i^k, e_{ij}^{k'})\right) + \lambda \cdot CEAScore_{sub}(e_i^k)\right\} \tag{13}$$

Columns Property Annotation The set of candidate properties is denoted by $\mathscr{P}\{p \mid (e_i^G, hasProperty, p) \in KG, i = 1, 2 \ldots, m\}$. We assign a score to each property p in $\mathscr{P}$ with respect to the j-th column by:

$$I(p, e_i^k) = \begin{cases} M(e_i^k, e_{ij}^{k'}), & (e_i^k, p, e_{ij}^{k'}) \in KG \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

The CPA matching score is calculated by Eq.15.

$$CPAScore(p_j) = \sum_{i=1}^{M} \max_{k=1}^{N(c_i)} \left\{\phi\left(I(p_j, e_i^k)\right) + \lambda \cdot CEAScore_{sub}(e_i^k)\right\} \tag{15}$$

2. Results

In the Accuracy Track of SemTab 2022, participants compete with each other for three rounds. In each round, different datasets are provided to evaluate their systems on CTA, CEA, and CPA tasks.

Table.1 shows the results of KGCODE-Tab in all datasets of SemTab 2022. Since our system evolved as the competition went on, its rank and performance were on the rise during the whole competition.

Dataset \ Task	CTA			CEA			CPA		
	APrecision	AF1	Rank	APrecision	AF1	Rank	APrecision	AF1	Rank
Round1									
HardTablesR1(WD)	0.944	0.942	4	0.916	0.893	4	0.918	0.906	5
Round2									
HardTablesR2(WD)	0.971	0.968	1	0.875	0.856	2	0.943	0.916	3
ToughTables(WD)	0.546	0.543	1	0.913	0.905	3	/	/	/
ToughTables(DBP)	0.485	0.480	1	0.830	0.827	1	/	/	/
Round3									
BiodivTab(DBP)	0.867	0.867	1	0.911	0.911	1	/	/	/
GitTables(DBP)	0.608	0.587	2	/	/	/	/	/	/
GitTables(SCH)(class)	0.716	0.693	1	/	/	/	/	/	/
GitTables(SCH)(property)	0.665	0.618	2	/	/	/	/	/	/

Table 1
Results of KGCODE-Tab obtained in SemTab 2022.

2.1. Round 1

In Round 1, tables of HardTables datasets have small numbers of rows and columns, and the subject columns of most tables are the first columns. Thus KGCODE-Tab processes tables in batches and sets the first columns as subject columns by default. Experiments show that processing in batches dramatically improves the efficiency of spell correction and entity recall, fully utilizing the multithreading technology. Fixing subject columns also reduce the error caused by the table structure analysis module.

2.2. Round 2

In Round 2, the subject columns of tables in ToughTables datasets are not always the first columns, and non-subject columns are not necessary to be the properties of subject columns but can be their descriptions. Hence, the table structure analysis module comes into play, and the descriptions of entities participate in the calculation of similarity scores. Results show that these modifications largely increase the accuracy of the entity disambiguation module, improving the ranking of our system.

In addition, the number of rows in each table in ToughTables datasets fluctuates greatly, and some tables have extremely large numbers of rows. Hence, adaptive batch processing is introduced according to the size of the tabular data, and for the table with a large number of rows, only part of the representative rows are randomly selected for CTA task annotation, improving the efficiency of tabular data in spell correction and entity recall.

2.3. Round 3

In Round 3, tables in the BiodivTab datasets are about biodiversity, so KGCODE-Tab constructs a biodiversity corpus for abbreviations and aliases commonly used in the field of biodiversity. Furthermore, many cells contain noise like adverbs and adjectives, and most headers have semantic information. Therefore, tokenization is introduced to reduce the effect of noise, and KGCODE-Tab converts CTA task into CEA task for headers.

For Gittables datasets, by observing the annotation results of its training dataset, we find that the number of its labels is small and the type of annotation is relatively general, so we

consider using a text classification algorithm to solve the problem. After preliminary analysis and research, we select the FastText [12] model. Firstly, original words are divided into several tokens, and the CTA results are used as labels. Then the *spaCy* is used for word recognition, and the results are used as keywords. They are put into the FastText model for training. After training, it is used to annotate the test dataset.

3. General comments

In SemTab 2022, our KGCODE-Tab team participating in SemTab for the first time has a good result. Among all the participating teams, we achieve first-place results in multiple tasks.

KGCODE-Tab has some strategies to improve performance with less query time. The task analysis of the top layer can directly call the interface of the bottom layer, which increases the maintainability of the system. The tabular data preprocessing module makes full use of several tools like search engines, KGs API, and *spaCy* library to generate structured JSON files for each tabular data to increase reusability. To achieve the semantic annotation of tabular data, three tasks of CEA, CTA, and CPA are closely combined to deal with. As a whole, KGCODE-Tab fully utilizes the context of the whole table and the information provided by KGs to achieve a high accuracy.

However, the entity disambiguation module can continue to be optimized, and machine learning algorithms can be used to train parameters.

4. Conclusion

In this paper, we propose a novel table annotation system, KGCODE-Tab, which can deal with three TDKGM tasks: CTA, CEA, and CPA. We propose several effective tabular data preprocessing techniques, which consist of table structure analysis, spell correction, and entity recall. KGCODE-Tab emphasizes entity disambiguation with table context, which reduces much noise and remains candidate entities with high confidence. For each task, we design a scoring formula to select the right answer among candidate entities, which utilizes the results from other tasks. Results of SemTab 2022 show that KGCODE-Tab has excellent disambiguation ability and achieves outstanding performance.

Supplemental Material Statement: Source code and constructed datasets will be released on GitHub soon.

References

[1] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems, in: Proceedings of the 17th Extended Semantic Web Conference (ESWC 2020), Berlin, Heidelberg, 2020.

[2] R. Azzi, G. Diallo, Amalgam: Making tabular dataset explicit with knowledge graph, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph

Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual, Online, 2020.

[3] S. Chen, A. Karaoglu, C. Negreanu, T. Ma, J.-G. Yao, J. Williams, A. Gordon, C.-Y. Lin, Linkingpark: An integrated approach for semantic table interpretation, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual, Online, 2020.

[4] Y. Chabot, T. Labbe, J. Liu, R. Troncy, Dagobah: An end-to-end context-free tabular data semantic annotation system, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New zealand, 2019.

[5] V.-P. Huynh, J. Liu, Y. Chabot, T. Labbe, P. Monnin, R. Troncy, Dagobah: Enhanced scoring algorithms for scalable annotations of tabular data, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual, Online, 2020.

[6] S. Yumusak, Knowledge graph matching with inter-service information transfer, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual, Online, 2020.

[7] D. Oliveira, M. d'Aquin, Adog - annotating data with ontologies and graphs, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New zealand, 2019.

[8] M. Cremaschi, R. Avogadro, D. Chieregato, Mantistable: An automatic approach for the semantic table interpretation, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New zealand, 2019.

[9] H. Morikawa, Semantic table interpretation using lod4all, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New zealand, 2019.

[10] B. Steenwinckel, G. Vandewiele, F. de Turck, F. Ongenae, Csv2kg: Transforming tabular data into semantic knowledge, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2019) co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New zealand, 2019.

[11] S. Tyagi, E. Jimenez-Ruiz, Lexma: Tabular data to knowledge graph matching using lexical techniques, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual, Online, 2020.

[12] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, 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, 2017.