Semantic Annotation of Tabular Data for Machine-to-Machine Interoperability via Neuro-Symbolic Anchoring

Shervin Mehryar[1,*,†], Remzi Celebi[1,†]

[1]Institute of Data Science, Maastricht University, Paul-Henri Spaaklaan 1, 6229 GT, Maastricht, Netherlands

Abstract

In this paper we investigate automated annotation of tabular data using semantic technologies in combination with neural network embedding. Specifically, we propose an anchoring model in which property and cell types from the data embedding space are aligned with ontology relation and entity types. We show that by combining the power of symbolic reasoning, neural embeddings, and loss function design, a significant performance improvement as high as 86% for column property, 82% for column type, and 87% for column qualifier annotations can be achieved based on DBpedia and Wikidata table extractions.

Keywords

Semantic Annotation, Tabular data, Neuro-symbolic AI, Interoperability

1. Introduction

Structuring data in tabular format is a common method for information processing systems across many domains including health care [1], law and tech [2, 3], and food industry [4]. Annotating the tabular content with the semantic types (i.e., the semantic types of columns, column values and their relationship) is a crucial step in making such data interoperable among institutions and their users. Dealing with real tabularly formatted data presents several syntactic and semantic issues that make the annotation task particularly difficult. First and foremost, the column types, which indicate the type of each corresponding cell, are often unknown beforehand. Although column headers may provide some information, there might be no direct connection to an ontology term, which complicates the annotation process. Another closely related issue is that the particular relation between column pairs may also be unknown a priori. This issue is known as property type matching. Lastly, one might be interested in annotating the cells as entities with groundings from a knowledge graph. In certain cases, the relations can be further complicated when dealing with one-to-many and many-to-one cases (i.e., qualifier type prediction).

SemTab'23: Semantic Web Challenge on Tabular Data to Knowledge Graph Matching 2023, co-located with the 22nd International Semantic Web Conference (ISWC), November 6-10, 2023, Athens, Greece

*Corresponding author: shervin.mehryar@maastrichtuniversity.nl

shervin.mehryar@maastrichtuniversity.nl (S. Mehryar); remzi.celebi@maastrichtuniversity.nl (R. Celebi)

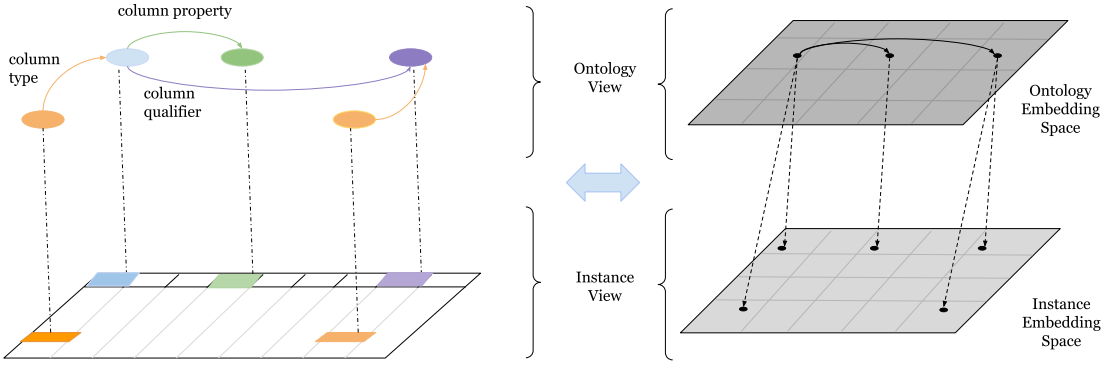

Figure 1: Neuro-symbolic vector spaces for Ontology and Table data: table entries are used as anchors with types determined from a source knowledge graph (**left**) and their embeddings maintained in alignment for consistency in the vector space representations (**right**).

In this paper we investigate automated annotation of tabular data using semantic technologies in combination with natural language embedding models. Our objective is to detect, interpret, and map column headers and values along with relationships from a tabular data in alignment with a source ontology. Most automated approaches for semantic annotation of tabular data rely either on symbolic methods (e.g., syntactic or semantic similarity), or use neural network classification where entities are represented with latent vectors. Neuro-symbolic reasoning [5, 6] is a new paradigm that aims to combine the inductive power of symbolic reasoning with the robustness of neural network predictions. In this work, we propose a neuro-symbolic method in order to achieve reliable and consistent semantic annotations of tabular data in a fully automated manner.

We present an algorithm named MUT2KG that models the relations between concepts from a knowledge graph (Ontology) to tabular data (schema) by learning and aligning embeddings in each respective space. To accomplish this, we extract column and property types from each table and map them to types in the knowledge graph during embedding learning. Furthermore, we consider the cell values as groundings of objects in a source knowledge graph (e.g. DBpedia or Schema.org). For learning embeddings in our vector space, we generate samples as triples from table entries with column labels as relations. In this process, the column types (from the knowledge graph embedding space) and the column labels (as properties in the entity embedding space) act as anchors, following a similar approach in [7]. These anchors ensure that data are properly aligned in both spaces. The true benefit of the proposed framework is in generating hiqh-quality embeddings for knowledge completion, whereby at test time the type of an entity or a property based on entries in a table can be inferred via the infused information in their vector embeddings.

2. Related Work

Knowledge graphs provide a rich source to symbolic methods for the semantic annotation of tabular data. In a knowledge graph, the facts are stored as triples and accessed through an appropriate query language. A system called SemInt [8] proposes to query knowledge graphs to address entity and property annotation tasks. It follows a majority-voting scheme to select the type of a column based on a candidate pool of types generated from querying the reference knowledge graph over the corresponding cells. JenTab [9] retrieves column types, relations and cell types based on an extensive and interlinked query systems. It further uses a maximum length ancestry path approach to perform annotation from a set of candidate types. We use a combination of query-based methods from SemInt and JenTab to extract facts/triples of the reference knowledge graphs for embedding purposes.

Recently, a number of deep learning models have aimed at solving the task of table annotation. In [10], the idea of pre-training on relational tables is introduced in an unsupervised manner using a structure-aware Transformer with a masked entity recovery mechanism. The pre-trained model is subsequently fine-tuned for specific tasks including entity linking, column type annotation, relation extraction, subject column entity population, object entity filling, and schema/header augmentation. In [11], a combination of word and paragraph embeddings together with table statistics (e.g. character distributions) as hints to the neural network is used to improve performance over decision-tree like models. We employ the same neural network based architectures as building blocks in entity representation and learning. There are approaches that resort to large language models for this application as well in an end-to-end fashion, but we leave that as future research direction.

There has been a rise in joint embedding learning and reasoning over knowledge graphs in recent years. A complete survey of these methods under the umbrella of neuro-symbolic AI is provided in [12]. In [13], a joint text query and database retrieval model is proposed that relies on a cell linearization technique (i.e. encoding entities with column type and other information) to provide answer to textual questions in a query-and-answer-type manner. Finally, a joint text and knowledge graph embedding method is proposed in [7] whereby the context provided by neural text processing improves the quality of learned embeddings through the knowledge graph, and vice versa. Similarly, we infuse the knowledge graph embeddings with context information provided by the tabular data in order to achieve higher performance.

3. Methodology

In this paper we propose a novel approach based on the groundings of concepts and relations from a source knowledge graph (e.g. DBpedia or Schema.org) to provide type annotations for a given table. Figure (1) shows the proposed framework. More specifically, we consider each table as a flattened version of a sub-graph sampled from a source knowledge graph. The table can be treated as a local (often noisy) extraction. In this vein, we represent relations among entities and columns similar to nodes and concepts in a knowledge graph. This view

subsequently allows the construction of an embedding space in which the entities from the local knowledge graph (i.e. flat mapping of table data) are aligned with the relevant part in the source knowledge graph, whereby entity and column type annotations become node and link predictions, respectively.

We formulate the problem of semantic annotation as an optimization problem for which vector embeddings are learned for entities and relations in a knowledge graph $\mathcal{G}$ jointly with entries and headers in a table $\mathcal{T}$. If $(h, p, t) \in \mathcal{G}$ represents a fact as a triple in the knowledge graph, then in the corresponding vector space the vector relation $\vec{h} + \vec{p} = \vec{t}$ holds. Alternately, for two cell entries e_1 and e_2 on one row, $(e_1, r, e_2) \in \mathcal{T}$ represents a triple (flat) mapping using column property r, for which the vector relation $\vec{e_1} + \vec{r} = \vec{e_2}$ holds. We hypothesize that for a grounding $e_1 = h$ and $e_2 = t$, the vectors $\vec{r}$ and $\vec{p}$ are similar. In other words, the knowledge graph predicate and table column property match. We further extend this idea by considering anchor nodes in each space, namely a_g and a_t, corresponding to entities in knowledge graph and entries in tabular data that are semantically similar. Next we define loss functions to capture these intuitions in order to learn high-quality embeddings. In particular, we define three sets of loss functions: capturing ontology-view triples and instance-view triples similar to [14] and a new loss term corresponding to anchoring triples that in essence perform alignment between the two representation spaces, neuro-symbolically.

Ontology-view Loss: given a set $\mathcal{G}$ of triplets representing facts in a knowledge graph, the ontology view loss refers to the following criterion:

$$L_{\mathcal{G}} = \sum_{(h,p,t)\in\mathcal{G}} ||\vec{h} + \vec{p} - \vec{t}||_2 \quad , \tag{1}$$

where as before, $\vec{h}$, $\vec{p}$, and $\vec{t}$ are vector representations in the vector space $\mathcal{G}$ corresponding to head, predicate, and tail entities in the source ontology. These representations are learned using triples in the ontology and embeded as a d-dimensional vector similar to the process in aTransE [15].

The set of triples $\mathcal{G}$ includes subjects, predicate, and objects from the source ontology. One approach in this case would be to embed the entire ontological graph, which may be computationally intractable as the size of the ontology grows (e.g. WikiData). Alternately, relevant parts of the entire knowledge graph can be embedded as needed. To this end, given a large, source ontology and a corresponding set of ontological grounded data (i.e. table values), the relevant information from the source ontology can be extracted by column type, entity type, and property type queries using 'rdf:type' and sub-class predicates. In particular, we employ a combination of queries run across the column and cell entities from the tabular data, in order to retrieve concept hierarchies reflecting the relevant ontological components in the data (see SemInt [8] and JenTab [9] for details). We note that, this process is not expected to be perfect due to noise, variations, missing information, and other sources of error in the data. However, it is robust in the sense that with enough data, a representative set can be obtained which will further be complemented throughout the learning process using the power of knowledge graph

embeddings for knowledge completion.

Instance-view Loss: this loss function captures the information directly contained in tables. In other words, it learns a set of embeddings for the cell and column entities, and the relation between them. Formally, let $\mathscr{T}$ be the set of triplets representing entities in a table, the instance view loss refers to the following criterion:

$$L_{\mathscr{T}} = \sum_{(e_1,r,e_2)\in\mathscr{T}} \|\vec{e_1} + \vec{r} - \vec{e_2}\|_2 \quad , \tag{2}$$

where as before, $\vec{e_1}$, $\vec{r}$, and $\vec{e_2}$ are vector representations in the vector space $\mathbb{R}^d$ corresponding to subject, relation, and object entities in the table data. Since the data is readily available, albeit noisy, in a 1-to-1 format, a translation based method such as TransE [16] can be used to learn local embeddings. However, TransE does not perform well as far as one-to-many and many-to-one relations are concerned. This is the case in qualifier based prediction tasks. Therefore, we extend this notion to cover such cases with a modification to the equation (2) as follows:

$$L'_{\mathscr{T}} = \sum_{(e_1,r,e_2)\in\mathscr{T}} \|\vec{e_1} + \vec{r} - \vec{e_2}\|_2 + \sum_{\substack{(s,r_1,t_1)\in\mathscr{T} \\ (s,r_2,t_2)\in\mathscr{T} \\ r_1\neq r_2}} \|\vec{t_1} - \vec{t_2} + \vec{r_2} - \vec{r_1}\|_2 \quad , \tag{3}$$

where the first term remains the same as previously and the second term now captures one-to-many relations as follows. Suppose an entity s is subject to two different relations r_1 and r_2 with target entities t_1 and t_2, i.e. (s, r_1, t_1) and (s, r_2, t_2). In vector space notation, this relationship can be denoted by $\vec{s} = \vec{t_1} - \vec{r_1}$ and $\vec{s} = \vec{t_2} - \vec{r_2}$. Setting the right-hand-side of the latter relations to be equal, results in the desired term in the loss function. Thus far in our treatment, we have relied on entity types and relations in data and ontology domains separately. In order to make the connection between the two, we utilize the connection between the header entities and their types with the ontology concepts and hierarchies via anchor nodes, as explained next.

Anchor Loss: we define anchor pairs $(a, e) \in \mathscr{G} \times \mathscr{T}$ which relate column types and ontology concepts, for the purpose of clustering column type embeddings close to their concept embeddings in the end. Formally, let $\mathscr{A}$ be the set of all anchor pairs, a column type for which a matching concept exists, then anchor loss refers to the following criterion [14]:

$$L_{\mathscr{A}} = \sum_{\substack{(a,e)\in\mathscr{A} \\ (a,e')\notin\mathscr{A}}} \|\vec{a} - \vec{e}\|_2 - \|\vec{a} - \vec{e'}\|_2 + \gamma \quad , \tag{4}$$

where the first term and second term capture distances corresponding to positive and negative examples, and γ is a radius hyper parameter around the anchor nodes. In practice, $\vec{a}$ is taken to be a non-linear affine transformation of an ontology concept $c \in \mathscr{G}$ through a transformation $\vec{a} = \sigma(\mathbf{W}\vec{c} + \vec{b})$, where $\mathbf{W} \in \mathbb{R}^{dxd}$ and $\vec{b} \in \mathbb{R}^d$ are learned parameters. Here $\sigma()$ is an added non-linearity such as the sigmoid function. It should be mentioned that for the case that a pre-determined alignment between column types and concepts exists, the matrix $\mathbf{W}$ can

trivially be set to the identity matrix.

Extensive experimentation in [15] has shown that explicitly adding the hierarchical reasoning information from the ontology to the training loss makes a significant improvement in the quality of learned embeddings. We adopt the same idea and add concepts upto 2-hops away to the loss function in equaiton (4) to further improve the anchoring effect as follows:

$$L'_{\mathscr{A}} = \sum_{\substack{(a,e)\in\mathscr{A},(a,e')\notin\mathscr{A} \\ (a,r,a')\in\mathscr{G}}} \left[\|\vec{a}-\vec{e}\|_2 - \|\vec{a}-\vec{e'}\|_2 + \|\vec{a'}-\vec{e}\|_2 - \|\vec{a'}-\vec{e'}\|_2\right] + \gamma \quad , \tag{5}$$

where (a, r, a') now represents a first order hierarchy relation between the ontology concepts such that a' is a super-class of the ontology concept a.

The overall training process involves generating and collecting triples from tabular entities, headers as column and property types, as well as ontological and heirarchical triples from the source knowledge graph. Once this data is collected as training input, the overall objective function to optimize and learn the various embeddings with is given as:

$$L_{total} = L_{\mathscr{G}} + L'_{\mathscr{T}} + L'_{\mathscr{A}} \quad , \tag{6}$$

with $\mathscr{G}$, $\mathscr{T}$, and $\mathscr{A}$ referring to the aforementioned ontology, tabular, and anchor data sets. We use a neural network optimizer and stochastic gradient decent like algorithm to optimize over the objective function and obtain the desired embedding representations. In particular, since the tabular data are often noisy, we use powerful, pretrained deep bi-directional transformers to encode their data, such as BERT or DistilBERT [17]. For the encoding of source knowledge graph embeddings, a simpler mechanism without pretraining is utilized. In other words, in our experiments the concept and anchor representations, for which no pre-processing from the source knowledge graph is required, are learned from scratch through a single layer, linear transformation. On the other, pretrained neural tokenizers are used for the data that is extracted from tables.

The overall architecture of the proposed methodology is shown in Figure 2, including the loss functions mentioned above. In particular, the source ontology is queried based on entities in each cell as well the column types as the 'instance of' relation for cell type and column type annotation. The relation between the subject column and the target columns are additionally queried for column property annotation. For each table, this results in a corresponding subgraph extracted from the source ontology, which consists of (h, p, t) triples that are embedded using the loss equation 1. The subject, relation, and target entities from each table are turned into (e_1, r, e_2) and separately embedded using the loss equation 2. The entities retrieved in the subgraph that match exactly with the values in the tabular data form the basis for the anchoring set, which are used to align the embedding spaces using the loss equation 4.

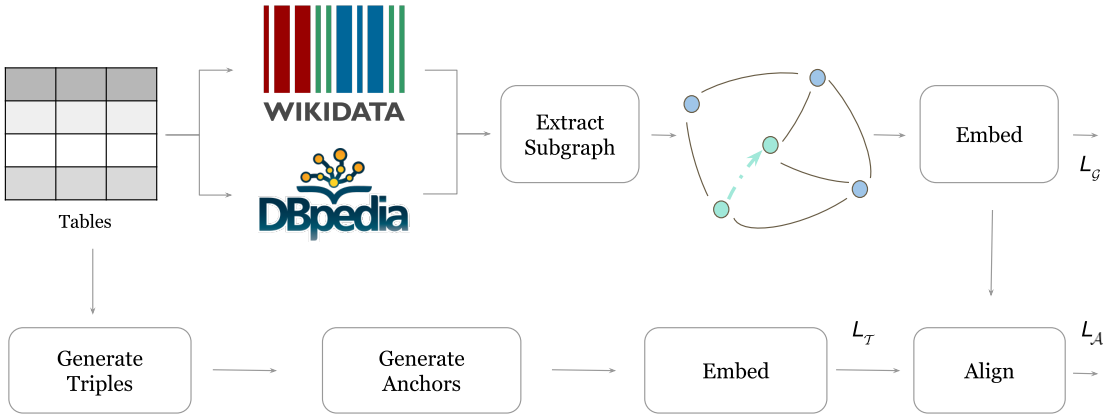

Figure 2: Entities in the tabulated data $\mathcal{T}$ are embedded as triples, using loss function $L_{\mathcal{T}}$, alongside triples extracted from the matching subgraph $\mathcal{G}$ generated from the source ontologies, using loss function $L_{\mathcal{G}}$. The alignment set $\mathcal{A}$ is determined by matching table and subgraph entity embeddings, using loss function $L_{\mathcal{A}}$.

4. Experiments and Results

For experiments, we use the tabular data from SemTab 2023 challenge[1] and report test results on three separate settings in Tables 1 and 2. In the first setting, we directly apply and report the query-based part of our approach on the provided Wikidata tables of round one, which includes a total of 769 tables with 3998 cell entities for cell type annotation, one subject column per table for column type annotation, and 298 non-subject columns from 401 of the tables for column property annotation. In the second setting, we use embedding/neural learning as explained in the previous section in addition to the types queried from the source ontologies (i.e. Schema.org and DBpedia), on a total of 44,407 tables provided for training in round 2. The challenge is posed as a multi-class classification problem in this setting with 44 and 80 types from DBpedia and Schema.org respectivaly for the column type annotation task, as well as with 49 and 103 types from from DBpedia and Schema.org respectively for the property type annotation task. Lastly, in the third setting a total of 844 tables are provided for training and performance is reported another dataset of 844 tables, for qualifier annotation.

The performance results using precision for the first setting/round are summarized in Table 1, where we only focus on query-based annotating. Specifically, in phase one a query using the 'instance of' predicate is performed to retrieve candidate column types for the subject column similar to [8]. Following that, in the second phase a set of property and entity candidates are retrieved with respect to each given target column. The set of candidates for subject column type, target column type, individual cell entities, and property types are further processed to compute the final results using a majority voting scheme similar to [9]. Since no parameter learning is required during this phase, the results are directly reported in terms of precision for cell entity at 0.587 (f1-score of 0.408), column type at 0.655 (f1-score of 0.459), and property

[1]Available: https://sem-tab-challenge.github.io/2023/

Table 1
Comparison of performance for Cell Entity Annotation (CEA), Column Type Annotation (CTA), and Column Property Annotation (CPA) tasks using WikiData tables. We compare the precision of our results MUT2KG-I against other three top methods from SemTab 2023, namely Semtex, Kepler-aSI, and TorchicTab. The best scores are bold faced and the second best are underlined.

Method		Kepler-aSI	Semtex	TorchicTab	MUT2KG-I
Wikidata	**CEA**	**0.959**	<u>0.904</u>	0.839	0.587
	CTA	0.739	**0.934**	<u>0.749</u>	0.655
	CPA	0.777	**0.968**	0.934	<u>0.948</u>

type at 0.94 (f1-score of 0.226) annotations using this approach, named MUT2KG-I.

The second set of test results are summarized in Table 2 for selected methods on the benchmark datasets and reported in terms of precision (a value between zero and one). Specifically, two main results are reported for column and property type annotations (CTA and CPA) using the method described in Section 3, based on two source ontologies (Schema.org and DBpedia). The results from our proposed method named MUT2KG-II, which now combines symbolic (query based) and neural anchoring, achieves a precision of 0.79 and 0.85 on the Schema.org dataset extraction for column and property annotation (with f1-scores of 0.32 and 0.79 respectively). On the DBpedia sets, the precision values are relatively higher, but not significantly so, with 0.82 and 0.86 for both CTA and CPA tasks (with f1-scores of 0.33 and 0.82 respectively). These come second to TorchicTac, but outperform TSOTSA and Anu.

Lastly, in the third set of results we focus on the qualifier prediction task which refers to predicting both main properties and qualifiers from Wikidata for n-ary relations. In particular, the focus is on binary relations where the property types between three given columns are to be predicted, with one having the role of a subject/pivot. For the qualifier annotation task on WikiData extractions [2], MUT2KG-II based on neuro-symbolic anchoring outperforms all other methods with 0.87 precision accuracy, seen in the last column in Table 2.

5. Discussions

As reported by the first setting the proposed method MUT2KG-I, where column headers and table entries are directly queried as column and property types from the source ontologies (i.e. Schema.org and DBpedia), achieves acceptable performance for the CEA and CTA tasks (see Table 1). This setting is equivalent to the approaches of SemInt and JenTab (see above) where repeated query and voting are used for candidate type selections. In particular, due to the generation of many candidates and its iterative nature, MUT2KG-I performance is well suited for the task of CPA with 0.948 precision. In the second setting the proposed method MUT2KG-II's performance, where entries and header columns are embedded using a bi-directional encoder trained separately with column headers as class labels and mapped

[2]Available: https://github.com/bennokr/semtab2023-CQA/

Table 2
Comparison of performance for Column Type Annotation (CTA), Column Property Annotation (CPA), and Column Qualifier Annotation (CQA) tasks using datasets based on Schema.org, DBpedia, and WikiData. We compare the precision of our results MUT2KG-II against other three top methods from SemTab 2023, namely TSOTSA, Anu, and TorchicTab. The best scores are bold faced and the second best are underlined.

Method	*SCHEMA Ontology Dataset*		*DBpedia Ontology Dataset*		*WikiData*
	CTA	**CPA**	**CTA**	**CPA**	**CQA**
TSOTSA	0.56	0.43	0.62	0.47	-
Anu	0.70	0.79	0.69	0.85	0.08
TorchicTab	**0.90**	**0.88**	**0.91**	**0.91**	<u>0.82</u>
MUT2KG-II	<u>0.79</u>	<u>0.85</u>	<u>0.82</u>	<u>0.86</u>	**0.87**

to the concepts in the source ontology, in the CTA and CPA tasks increases or remains high, respectively. This performance gain is deemed to be due to the added semantic enrichment, captured through the improvements introduced in equations 5 and 3 from the extracted subgraph. The effect is most pronounced in the CQA task with MUT2KG-II outperforming other methods over all the benchmarks (see Table 2).

One of the key components in the proposed method is the choice of an anchoring set introduced in equations 4 and 5, the role of which is to align the corresponding embedding vectors between the entities in the source ontology's extracted subgraph and the tabular instance data. In order to better characterize the effect of this set, an additional set of experiments is run in which the neural training and alignment of embeddings are performed only on a smaller subset of all matched anchor elements. These experiments are run with 10 difference random seeds and reports in terms of mean and standard deviation in Figure 2. In each run, between zero to one hundred percent of the anchoring set is used to learn the embeddings in the MUT2KG-II setting as before, for the CTA and CPA tasks using both Schema.org (SCH) and DBpedia (DBP) ontologies as source ontology.

For the task of column type annotation, it can be observed from the left plot that by using nearly %30 of the anchor set selected at random, the validation precision approaches the maximum achievable. Furthermore, this precision depends on the choice of the underlying ontology whereby the precision is increased through the use subgraphs extracted and embedded from DBP as opposed to SCH. In the low precision regime this difference is negligible. For the task of column property annotation as shown on the right, the precision steadily increases in either case to a peak near %40 of the anchor size and drops slightly thereafter. The reason for this behaviour could be partly explained by the relatively lower number of property types used across all tables, 49 properties from DBP and 103 properties from SCH, compared to the number of column types used as anchors in the CTA task, which shows after the inflection point. Once again, DBP used as the source ontology by MUT2KG-II results in better precision over SCH for subgraph matching and embedding.

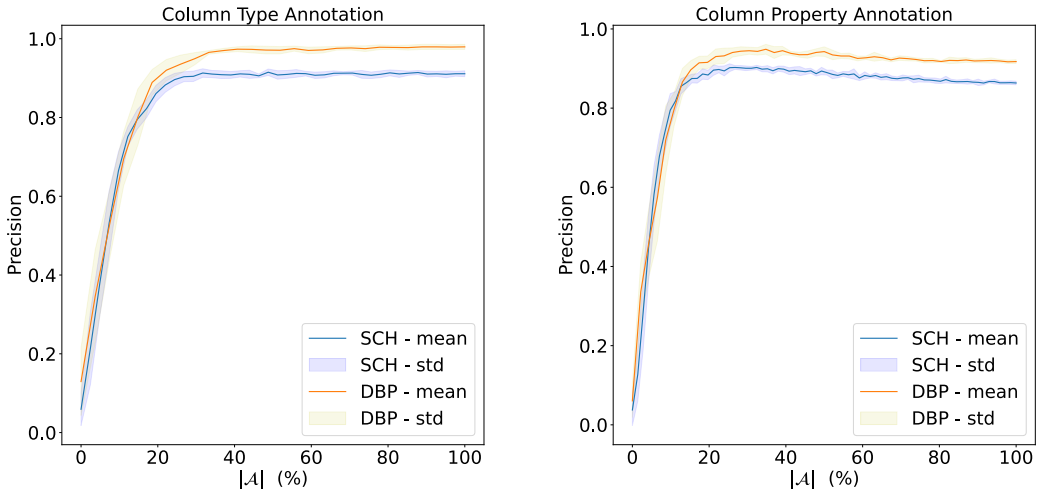

Figure 3: Precision (between 0 and 1) versus the size of the anchoring set $\mathscr{A}$ (in percentage) for the column type task (**left**) and column property task (**right**) using two source ontologies DBpedia (DBP) and Schema.org (SCH), shown as mean and standard deviation.

6. Conclusion

One of the main challenges of sharing data between institutions and their users in many domains such as clinical and legal applications is data interoperability due to non-standard formatting, for instance table schemas in case of tabular data. In this paper we propose a framework that provides annotations for column and property types for a given table in alignment with a source ontology. Specifically, we propose an anchoring model in which property and cell types from the data embedding space are aligned with ontology relation and entity types. Our method, evaluated on table extracts from SemTab 2023 challenge with Schema.org and DBpedia source ontologies achieves %86 for column property, %82 for column type, and %87 for column qualifier annotations.

Acknowledgments

The authors thank the challenge organizers for their timely and helpful response to inquiries, and the reviewers for their valuable comments. This work is supported by the Horizon Europe Framework Program (Grant agreement No: 101057062), project AIDAVA (AI powered Data Curation & Publishing Virtual Assistant).

References

[1] J. Chu, J. Chen, X. Chen, W. Dong, J. Shi, Z. Huang, Knowledge-aware multi-center clinical dataset adaptation: Problem, method, and application, Journal of Biomedical Informatics 115 (2021) 103710.

[2] R. Bakker, R. A. van Drie, M. de Boer, R. van Doesburg, T. van Engers, Semantic role labelling for dutch law texts, in: Proceedings of the Thirteenth Language Resources and Evaluation Conference, 2022, pp. 448–457.

[3] A. Louis, G. van Dijck, G. Spanakis, Finding the law: Enhancing statutory article retrieval via graph neural networks, in: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 2761–2776. URL: https://aclanthology.org/2023.eacl-main.203.

[4] A. Yilmaz, R. Naidu, C. Brewster, Fso: Food safety monitoring ontology (2023).

[5] J. Zhang, B. Chen, L. Zhang, X. Ke, H. Ding, Neural, symbolic and neural-symbolic reasoning on knowledge graphs, AI Open 2 (2021) 14–35.

[6] M. van Bekkum, M. de Boer, F. van Harmelen, A. Meyer-Vitali, A. t. Teije, Modular design patterns for hybrid learning and reasoning systems: a taxonomy, patterns and use cases, Applied Intelligence 51 (2021) 6528–6546.

[7] P. Rosso, D. Yang, P. Cudre-Mauroux, Revisiting text and knowledge graph joint embeddings: The amount of shared information matters!, in: 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019, pp. 2465–2473.

[8] A. Sharma, S. Dalal, S. Jain, Semint at semtab 2022, Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab), CEUR-WS. org (2022).

[9] N. Abdelmageed, S. Schindler, Jentab: Matching tabular data to knowledge graphs., in: SemTab@ ISWC, 2020, pp. 40–49.

[10] X. Deng, H. Sun, A. Lees, Y. Wu, C. Yu, Turl: Table understanding through representation learning, 2020. arXiv:2006.14806.

[11] M. Hulsebos, K. Hu, M. Bakker, E. Zgraggen, A. Satyanarayan, T. Kraska, Çağatay Demiralp, C. Hidalgo, Sherlock: A deep learning approach to semantic data type detection, 2019. arXiv:1905.10688.

[12] L. N. DeLong, R. F. Mir, M. Whyte, Z. Ji, J. D. Fleuriot, Neurosymbolic ai for reasoning on graph structures: A survey, 2023. arXiv:2302.07200.

[13] P. Yin, G. Neubig, W. tau Yih, S. Riedel, Tabert: Pretraining for joint understanding of textual and tabular data, 2020. arXiv:2005.08314.

[14] J. Hao, M. Chen, W. Yu, Y. Sun, W. Wang, Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts, 2021. arXiv:2103.08115.

[15] S. Mehryar, R. Celebi, Improving transitive embeddings in neural reasoning tasks via knowledge-based policy networks (2022).

[16] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Advances in neural information processing systems 26 (2013).

[17] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108 (2019).