RACOON: An LLM-based Framework for Retrieval-Augmented Column Type Annotation with a Knowledge Graph

Lindsey Linxi Wei, Guorui Xiao, Magdalena Balazinska
University of Washington
{linxiwei, grxiao, magda}@cs.washington.edu

Abstract

As an important component of data exploration and integration, Column Type Annotation (CTA) aims to label columns of a table with one or more semantic types. With the recent development of Large Language Models (LLMs), researchers have started to explore the possibility of using LLMs for CTA, leveraging their strong zero-shot capabilities. In this paper, we improve on LLM-based methods for CTA by showing how to use a Knowledge Graph (KG) to augment the context information provided to the LLM. Our approach, called **RACOON**, combines both pre-trained parametric and non-parametric knowledge during generation to improve LLMs' performance on CTA. Our experiments show that **RACOON** achieves up to a 0.21 micro F-1 improvement compared against vanilla LLM inference.

1 Introduction

In recent years, researchers and data scientists have found abundant relational tables on the Web (Cafarella et al. [2008a,b], Bhagavatula et al. [2015]). Despite their high quality, these Web Tables often miss important meta information such as column headers and relationships between columns (Suhara et al. [2022]). This information is essential for downstream tasks such as data quality control (Schelter et al. [2018]) and data discovery (Chapman et al. [2020]). A key step in recovering this meta information is *Column Type Annotation* (CTA), which aims to assign one or more *semantic type(s)* to columns in a given table. For example, the column ['UEFA Champions League', 'Scottish Cup'] has the annotation "time.event".

Thus, there has been increasing interest and effort recently in developing methods that can *automatically* assign semantic labels to columns (Deng et al. [2022], Suhara et al. [2022], Miao and Wang [2023]). Recent work (Kayali et al. [2024], Feuer et al. [2023]) argues that relying on *Pre-trained Language Models* (PLMs) is not sufficient because they need task-specific and dataset-specific fine-tuning with *carefully labeled training dataset* to achieve reasonable results. On the other hand, the recent development of *Large Language Models* (LLMs) (Radford et al. [2018], Brown [2020], Ouyang et al. [2022]) with pre-trained parametric memory has opened the possibility of bridging this gap by requiring minimal or even no training examples to achieve promising performance.

There has been much research effort in developing effective solutions that utilize LLMs for CTA and table related tasks (Narayan et al. [2022], Tian et al. [2024]). For CTA specifically, one line of work (Korini and Bizer [2023]) directly applies LLMs to solve CTA with promising performance. CHORUS (Kayali et al. [2024]) and Archetype (Feuer et al. [2023]) further develop other components using history and remapping to improve performances. Table-GPT (Li et al. [2024]) *fine-tunes* LLMs to adapt to relational data format to improve performance on various table understanding tasks.

At the core of existing methods, they directly apply LLMs without any external knowledge. Despite promising performance, LLM-based methods still encounter challenges dealing with outdated knowl-

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

edge (He et al. [2022]), producing factual inaccuracies (Ji et al. [2023]), and handling domain-specific or specialized queries (Kandpal et al. [2023]). To alleviate such problems, *Retrieval-Augmented Generation* (RAG), which enhances LLMs' generation by retrieving knowledge from external sources as non-parametric memory (Asai et al. [2023]), has emerged as a promising approach. Among the many external knowledge sources, *Knowledge Graphs* (KGs), can provide succinct yet informative knowledge to help analyze the intricate semantics of entities (Ji et al. [2021], Wang et al. [2017]). Specifically, KGs represent information in the form of entities (nodes) and relations (edges) (Wang et al. [2017]). Each edge is a factual triplet in the form of (subject, relation, object), making KG a well-structured data source. Many recent approaches have studied integrating KGs with LLMs/PLMs through RAG to improve performances on various tasks (Xu et al. [2020], Pan et al. [2024], Dehghan et al. [2024], Sun et al. [2024], Wen et al. [2024]) and saw promising performance gains compared with vanilla LLM inference. But none of the above approaches focuses on CTA. The most relevant work KGLink (Wang et al. [2024]) incorporates an external KG with PLM *during training* for CTA, thus encountering the exact PLM problems we mentioned before, whereas our work aims to leverage LLMs and KG to avoid expensive training or fine-tuning over large models.

Inspired by these advances, we argue that we can similarly bring external non-parametric knowledge to LLMs for CTA to improve performance and propose our framework **RACOON**. As shown in Figure 1, compared with vanilla LLM inference on the left, **RACOON** on the right unifies LLM inference with a KG by performing *three additional steps*: namely after ingesting a table as the input, **RACOON** first (1) looks at the column cells' entity mention and **retrieves** related information from a KG. However, this retrieved content cannot be directly applied due to its large size and noise. Thus **RACOON** needs to (2) **process** the retrieved content, and finally (3) serialize the compressed context to **augment** the prompt for the LLM. Note that **RACOON** is orthogonal to existing LLM-based methods for CTA, which enhance LLMs' performance through additional demonstrations and response post-processing.

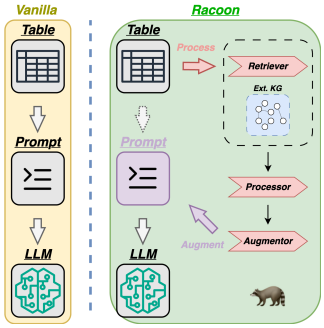

Figure 1: Vanilla way of using LLM for CTA (left) vs **RACOON** (right).

In summary, this paper makes the following contributions:

(1) We introduce the problem of augmenting prompts for LLMs on CTA with external knowledge from a KG and propose an end-to-end framework **RACOON** for it.

(2) We explore different granularity of information **RACOON** can retrieve from a KG with RAG. Furthermore, we provide effective post-retrieval compression and serialization methods such that given a column of cells, we can derive additional context information to augment the prompt.

(3) We conduct experiments showing that **RACOON** consistently outperforms vanilla LLM inference across various scenarios and retrieval methods, achieving up to a 0.21 improvement in micro F1.

2 Racoon's Approach

We now describe the **RACOON** framework. The overall workflow is shown in Figure 2. Besides a CTA query, **RACOON** also requires an external KG and an LLM. Note that **RACOON** treats the KG and LLM as black-boxes and does not require any modifications, thus enjoying a plug-and-play property. For clarity of presentation, we describe how **RACOON** works with a CTA query on a single column, but **RACOON** can process CTA queries on an arbitrary number of columns.

Retriever. Inside the Retriever, the *Parser* first parses the input table based on the query. Given a table T with a column Col_M to perform single-column CTA, the Parser selects C cells from Col_M for retrieval purposes. As of now, **RACOON** uses all cells, and we leave the problem of selecting the most representative cells for future work. The selected C cells are treated as text strings and passed to the *KG-Linker*, which takes entity mentions (the text strings) as input, connects with an external KG to link each entity mention to its referent entity in the KG, and returns the IDs of the referent entities. With those entity IDs, the final step is to further retrieve all entity labels and their one-hop neighbors in the KG as the output of the Retriever. The task KG-Linker performs is defined as Entity Linking (EL), a popular task in table interpretation (Han et al. [2011], Deng et al. [2022]). Since many EL methods are available online, **RACOON** allows users to implement their own KG-Linker based on the

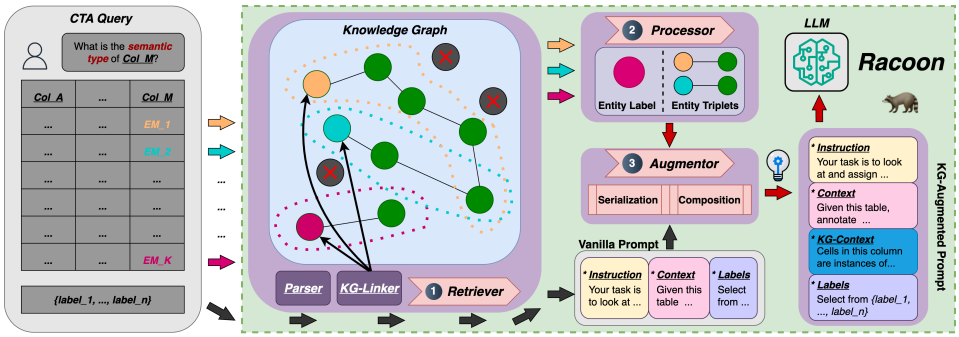

Figure 2: Overall **RACOON** workflow. **RACOON** performs CTA in a column-by-column manner while keeping the whole table for reference. The **Retriever** finds matching entities in the KG based on entity mentions in column cells and retrieves all relevant knowledge from KG. **Processor** reduces the retrieved cell-level knowledge into an *aggregated* column-level context. The compressed information is passed to **Augmentor**, which serializes all information into natural language and composes the KG-Context to form the final KG-Augmented Prompt.

KG they decided to use. We have experimented with two KG-Linkers: a vanilla API and a SOTA EL model (Ayoola et al. [2022]). Note that the Retriever assumes the KG-Linker can find entities in the connected KG. If it fails to link any selected cells, **RACOON** will fall back to using a vanilla prompt.

Processor. The role of the Processor is to compress and refine the retrieved information from the KG, ensuring it is both relevant and concise. The retrieved information takes the form of sets of nodes and edges for each selected cell in Col_M, which is too large and noisy for LLMs to digest. Thus, the Processor takes them as input and outputs their concise representations. As of now, **RACOON** supports processing the retrieved information into two representations, ENTITY-LABELS and ENTITY-TRIPLETS. ENTITY-LABELS include labels of entities in the KG. The intuition is that these labels are canonical representations of entity mentions, which are often ambiguous or incomplete. For example, the entity mention '15 Sge' is linked to the entity '15_Sagittae' in Wikidata KG, making it easier for LLMs to understand. Although ENTITY-LABELS can disambiguate entity mentions, they do not provide any further information about the entities such as their types. ENTITY-TRIPLETS, on the other hand, include triplets in the format (subject, relation, object). In our experiments, the linked entity serves as the subject, the relation is the *instance_of* relation, and the object is the entity type of which the subject is an instance. For example, (15_Sagittae, instance_of, star). Note that ENTITY-LABELS is strictly a subset of ENTITY-TRIPLETS. With this representation, **RACOON** fully leverages the ontological relationships among entities, enabling a deeper understanding of their types. **RACOON** then uses statistics to count occurrences of each entity node and relationship triplet for all selected cells in Col_M to give summary *column-level* information, resembling the compress stage in traditional RAG systems.

Augmentor. Finally, the Augmentor takes the original vanilla prompt and the processed representations from the Processor to create the KG-augmented prompt. It serializes the summarized column-level KG information into natural sentences and creates the KG-Context to insert into the original vanilla prompt. We show the detailed prompt in Figure 3.

3 Experiments

Setup. We evaluate **RACOON** using the GPT-3.5 model (Ouyang et al. [2022]) and Wikidata KG (Vrandečić and Krötzsch [2014]) on the full test set of the WikiTables-TURL-CTA benchmark (Deng et al. [2022]) consisting of 13,025 columns extracted and annotated in a multi-label manner using 255 Freebase's types by the TURL team from the WikiTable corpus. The dataset provides a ground truth entity ID (Wikipedia page ID) linked to each cell in the table. On average, tables in this dataset have a mean of 21 rows and 4 entity columns (columns with at least one linked cell).

Baselines. We compare **RACOON** against a vanilla LLM method on CTA. Notably, both CHORUS and ArcheType (Kayali et al. [2024], Feuer et al. [2023]) applies such vanilla LLM inference.

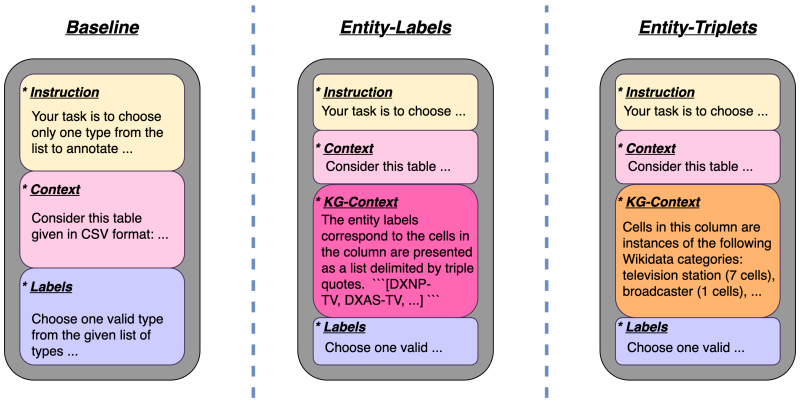

Figure 3: Vanilla (left) and **RACOON** (middle and right) prompts. *Instruction* describes the CTA task in natural language, *Context* describes the serialized table, and *Labels* shows the label set. The middle prompt uses ENTITY-LABELS and the right prompt uses ENTITY-TRIPLETS.

Table 1: The results of the baseline and **RACOON** in micro-F1 scores

		Multi-label			Single-label		
KG-Linker	EL	Baseline	ETL	ETT	Baseline	ETL	ETT
Ground Truth	1.0000	0.2609	0.4023	0.4500	0.4631	0.564	0.6814
MediaWiki API	0.4018	0.2609	0.3582	0.3494	0.4631	0.4927	0.5222
ReFinED	0.6255	0.2609	0.3698	0.3678	0.4631	0.4900	0.5705

Evaluation. We first use the ground truth entity ID linked to each cell labeled in the WikiTables-TURL-CTA dataset as the output of the KG-Linker to retrieve information from the KG. This enables us to evaluate our approach assuming a perfect KG-Linker. The results in Table 1 show that both augmented prompts using ENTITY-LABELS (ETL) and ENTITY-TRIPLETS (ETT) outperform vanilla LLM inference, with ENTITY-TRIPLETS achieving the best performance overall. This highlights the potential of retrieving more comprehensive information about column cells from the KG.

Second, we evaluate **RACOON**'s performance with different KG-Linkers: MediaWiki API (with action=websearchentities) and a SOTA entity linking model, ReFinED (Ayoola et al. [2022]). With either KG-linker, Racoon continues to outperform the baseline, although its performance (both ENTITY-LABELS and ENTITY-TRIPLETS) drops compared with the ground-truth linker and the drop is higher for ENTITY-TRIPLETS, showing that collecting extended information is not as helpful when the initial linked entity is incorrect. The drop is less with the ReFinED KG-linker, showing that SOTA KG-linkers suffice to make our approach beneficial.

Because LLMs often underperform in the multi-label dataset setting due to the large number of possible answer combinations, we further test **RACOON** in the single-label setting: If the model prediction falls within the ground truth label set, we set the ground truth label to be the model prediction. Otherwise, we count it as an incorrect prediction. In this setting, the performance of both the baseline and **RACOON** improves. **RACOON** with ReFinED significantly outperform the baseline, with ENTITY-TRIPLETS giving the best results.

4 Conclusion

We presented a novel end-to-end LLM-based framework **RACOON** for Column Type Annotation (CTA) unifying vanilla LLM and Knowledge Graph (KG) augmented inference. **RACOON** has three components: a Retriever for retrieving knowledge from a KG; a Processor for post-processing the retrieved content; and an Augmentor for composing the final prompt. Experiments show that **RACOON** *consistently* outperforms vanilla LLM for CTA.

Acknowledgments

This project was funded in part by a gift from NEC.

References

Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, 1(1):538–549, 2008a.

Michael J Cafarella, Alon Y Halevy, Yang Zhang, Daisy Zhe Wang, and Eugene Wu. Uncovering the relational web. In *WebDB*, pages 1–6. Citeseer, 2008b.

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. Tabel: Entity linking in web tables. In *International Semantic Web Conference*, pages 425–441. Springer, 2015.

Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. Annotating columns with pre-trained language models. In *Proceedings of the 2022 International Conference on Management of Data*, pages 1493–1503, 2022.

Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Biessmann, and Andreas Grafberger. Automating large-scale data quality verification. *Proceedings of the VLDB Endowment*, 11(12):1781–1794, 2018.

Adriane Chapman, Elena Simperl, Laura Koesten, George Konstantinidis, Luis-Daniel Ibáñez, Emilia Kacprzak, and Paul Groth. Dataset search: a survey. *The VLDB Journal*, 29(1):251–272, 2020.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1):33–40, 2022.

Zhengjie Miao and Jin Wang. Watchog: A light-weight contrastive learning based framework for column annotation. *Proc. ACM Manag. Data*, 1(4), dec 2023. doi: 10.1145/3626766. URL `https://doi.org/10.1145/3626766`.

Moe Kayali, Anton Lykov, Ilias Fountalis, Nikolaos Vasiloglou, Dan Olteanu, and Dan Suciu. Chorus: Foundation models for unified data discovery and exploration. *Proc. VLDB Endow.*, 17 (8):2104–2114, may 2024. ISSN 2150-8097. doi: 10.14778/3659437.3659461. URL `https://doi.org/10.14778/3659437.3659461`.

Benjamin Feuer, Yurong Liu, Chinmay Hegde, and Juliana Freire. Archetype: A novel framework for open-source column type annotation using large language models. *arXiv preprint arXiv:2310.18208*, 2023.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Tom B Brown. Language models are few-shot learners. *arXiv preprint ArXiv:2005.14165*, 2020.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Avanika Narayan, Ines Chami, Laurel Orr, Simran Arora, and Christopher Ré. Can foundation models wrangle your data? *arXiv preprint arXiv:2205.09911*, 2022.

Yuzhang Tian, Jianbo Zhao, Haoyu Dong, Junyu Xiong, Shiyu Xia, Mengyu Zhou, Yun Lin, José Cambronero, Yeye He, Shi Han, et al. Spreadsheetllm: Encoding spreadsheets for large language models. *arXiv preprint arXiv:2407.09025*, 2024.

Keti Korini and Christian Bizer. Column type annotation using chatgpt. *arXiv preprint arXiv:2306.00745*, 2023.

Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. Table-gpt: Table fine-tuned gpt for diverse table tasks. *Proc. ACM Manag. Data*, 2(3), may 2024. doi: 10.1145/3654979. URL https://doi.org/10.1145/3654979.

Hangfeng He, Hongming Zhang, and Dan Roth. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*, 2022.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR, 2023.

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pages 41–46, 2023.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514, 2021.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE transactions on knowledge and data engineering*, 29(12): 2724–2743, 2017.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. Controllable story generation with external knowledge using large-scale language models. In *Conference on Empirical Methods in Natural Language Processing*, 2020. URL https://api.semanticscholar.org/CorpusID:222125036.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

Mohammad Dehghan, Mohammad Alomrani, Sunyam Bagga, David Alfonso-Hermelo, Khalil Bibi, Abbas Ghaddar, Yingxue Zhang, Xiaoguang Li, Jianye Hao, Qun Liu, Jimmy Lin, Boxing Chen, Prasanna Parthasarathi, Mahdi Biparva, and Mehdi Rezagholizadeh. EWEK-QA : Enhanced web and efficient knowledge graph retrieval for citation-based question answering systems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14169–14187, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.764.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nnVO1PvbTv.

Yilin Wen, Zifeng Wang, and Jimeng Sun. MindMap: Knowledge graph prompting sparks graph of thoughts in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10370–10388, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.558.

Y. Wang, H. Xin, and L. Chen. Kglink: A column type annotation method that combines knowledge graph and pre-trained language model. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1023–1035, Los Alamitos, CA, USA, may 2024. IEEE Computer Society. doi: 10.1109/ICDE60146.2024.00083. URL https://doi.ieeecomputersociety.org/10.1109/ICDE60146.2024.00083.

Xianpei Han, Le Sun, and Jun Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774, 2011.

Tom Ayoola, Shubhi Tyagi, Joseph Fisher, Christos Christodoulopoulos, and Andrea Pierleoni. ReFinED: An efficient zero-shot-capable approach to end-to-end entity linking. In Anastassia Loukina, Rashmi Gangadharaiah, and Bonan Min, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 209–220, Hybrid: Seattle, Washington + Online, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-industry.24. URL `https://aclanthology.org/2022.naacl-industry.24`.

Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

A Appendix

Here we show the example prompts for the single-label setting and the multi-label setting.

A.1 Single-label

System message: Be a helpful, accurate assistant for data discovery and exploration desiged to output valid JSON in the format {'type': []}

User message: Consider this table given in Comma-separated Values format: "' Table "' There are a list of 255 valid types for each column: types. Your task is to choose only one type from the list to annotate the first column. Solve this task by following these steps: 1. Look at the cells in the first column of the above table. 2. Consider this information carefully: Cells in this column are instances of the following wikidata entities: human (6 cells). 3. Choose only one valid type from the given list of types. Check that the type MUST be in the list. Give the answer in valid JSON format.

A.2 Multi-label:

System message: Be a helpful, accurate assistant for data discovery and exploration desiged to output valid JSON in the format {'type': []}

User message: For multi-label: Consider this table given in Comma-separated Values format: "' Table "' There are a list of 255 valid types for each column: types. Your task is to choose one or multiple types from the list to annotate the first column. Solve this task by following these steps: 1. Look at the cells in the first column of the above table. 2. Consider this information carefully: Cells in this column are instances of the following wikidata entities: human (6 cells). 3. Mark each type in the given list with 0 or 1. Mark a type with 1 if it can better represent all cells of the first column. Mark a type with 0 otherwise. 4. Give a list of types that you have marked 1 in the previous step. Check that the types MUST be in the list. Give the answer in valid JSON format.