Knowledge Graph Entity Linking via Interactive Reasoning and Exploration with GRASP

Sebastian Walter[1,*], Hannah Bast[1]

[1]University of Freiburg, Georges-Köhler-Allee 51, 79110 Freiburg im Breisgau, Germany

Abstract

Entity linking is the task of linking mentions of entities from a given knowledge graph in a given text. We present a new approach for entity linking built on GRASP, a zero-shot method originally developed for translating natural-language questions to SPARQL queries for a given knowledge graph. We evaluate our approach on the cell entity annotation task of the MammoTab track of the SemTab 2025 challenge and achieve an F_1-score of 75.8%, ranking first among all participants.

Keywords

Entity Linking, Large Language Model, Knowledge Graphs, Cell Entity Annotation

1. Introduction

Entity linking is the task of linking mentions of entities from a given knowledge graph in a given text. It is a fundamental task in natural language processing because semantically enriching text in this way is useful for a wide variety of applications [1]. The classical approach to entity linking proceeds in two stages: First, identify passages in the text that mention an entity from the knowledge graph; this is known as *entity recognition*. Second, identify which of the entities from the knowledge graph is being referred to, if any; this is known as *entity disambiguation*. More recent neural methods try to solve the task end-to-end, either using a sequence-labeling approach (given the text, identify the mentions and annotations in a single forward pass) or using a sequence-to-sequence approach (given the text, generate a variant of the text with annotations) [2]. For a fair and in-depth performance evaluation of existing entity linkers, see [3].

Our solution goes one step further by using an agentic approach. Given the text and the knowledge graph, we use a large language model (LLM) to figure out the correct annotations interactively, by probing the knowledge graphs in various ways and reasoning with the results. This is a variant of an approach, called GRASP [4], that we have previously applied for translating natural-language questions to SPARQL queries on a given knowledge graph. Our approach is zero-shot, that is, does not require prior training on the knowledge graph or the given text. The overall process is very much inspired by how a human would go about this task.

Contributions

1. We present an approach for zero-shot knowledge graph entity linking from natural language inputs based on the interaction of an LLM with knowledge graphs through function calls (see Section 2).
2. We demonstrate the effectiveness of this approach on the cell entity annotation (CEA) task of the MammoTab track of the SemTab 2025 challenge [5], where we reach first place (see Section 3).
3. We integrate the CEA task into the GRASP system [6] available at github.com/ad-freiburg/grasp. We also publish our configuration files, scripts, and links to our predictions for the SemTab 2025 challenge in the semtab-2025 branch for full reproducibility of our results.

OM 2025: The 20th International Workshop on Ontology Matching co-located with the 24th International Semantic Web Conference (ISWC 2025), November 2nd, 2025, Nara, Japan

*Corresponding author.

swalter@cs.uni-freiburg.de (S. Walter); bast@cs.uni-freiburg.de (H. Bast)

https://ad.informatik.uni-freiburg.de/staff/walter (S. Walter); https://ad.informatik.uni-freiburg.de/staff/bast (H. Bast)

0009-0006-2613-3209 (S. Walter); 0000-0003-1213-6776 (H. Bast)

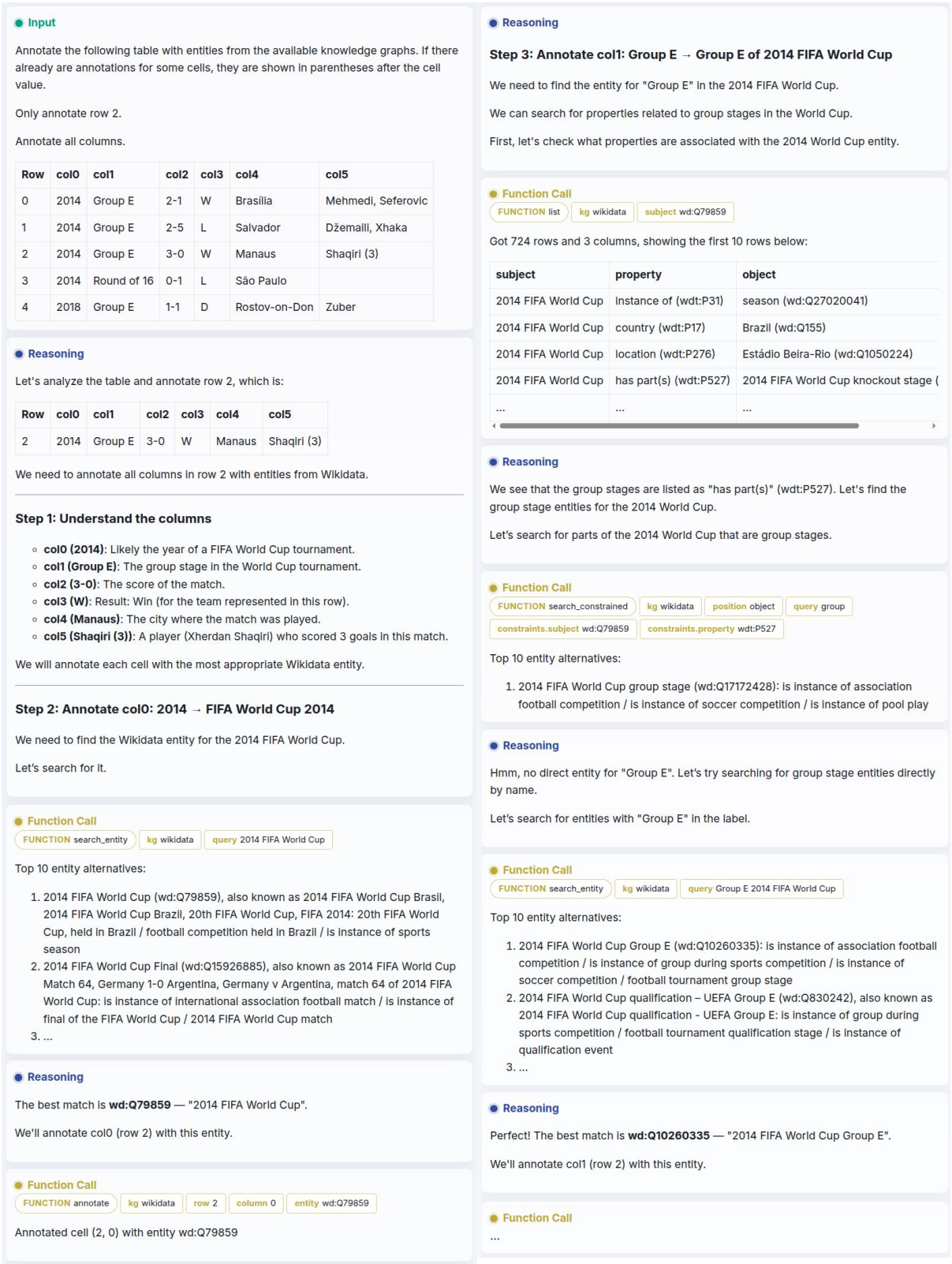


Figure 1: Excerpt of a cell entity annotation trace from running our approach with Qwen3 Next 80B A3B to annotate a particular row of a MammoTab table without any pre-existing annotations. In this example, only two rows of context are used and some of the intermediate function call results are truncated for brevity. See grasp.cs.uni-freiburg.de/share/jNR8xt for the full trace.

2. Approach

Our approach is based on GRASP, which equips an LLM with a set of functions to search and query knowledge graphs in an interactive fashion. Originally, GRASP was developed for *SPARQL QA* (translating natural-language questions to SPARQL queries on a knowledge graph); see [4] for the details of the method, and [6] for an implementation. In [6], the method was extended to support general QA and follow-up questions. In this work, we extend it to entity linking.

GRASP provides the following core functions, which allow the LLM to interactively explore knowledge graphs: EXE (execute an arbitrary SPARQL query), LST (list triples with given constraints), SEN (search for entities matching a given query string), SPR (search for properties matching a given query string), SPE (search for properties of a given entity), SOP (search for objects of a given property), SCN (search for items given triple constraints), and SAC (search for items given a constraining SPARQL query).

For the CEA task, we remove the SPARQL-QA-specific functions ANS (answer and stop) and CAN (cancel and stop), we keep the other functions, and we introduce the following four new functions:

`ANN: annotate(kg: str, entity: str, row: int, col: int)`
Add the given entity from the given knowledge graph as annotation for the table cell at the given row and column. If there already exists an annotation, it is overwritten.

`DAN: delete_annotation(row: int, col: int)`
Delete the annotation for the table cell at the given row and column.

`SAN: show_annotations()`
Show the current annotations.

`STP: stop()`
Stop the annotation process. The current annotations are returned as result.

Intuitively, GRASP tackles entity linking using the above functions in the following way: It identifies entity candidates using one of the various search functions, typically SEN. If multiple entities match well, it can disambiguate them by inspecting knowledge graph triples for each, typically using LST, or by performing more restrictive searches with constraints from already known entities or other patterns observed in the input, typically using SCN or SAC. If no entity matches well, it can retry different search queries or explore the knowledge graph starting from related entities in the hope to come across a match. If multiple attempts to find an appropriate entity fail, no annotation is made. In case there is an obvious relationship between multiple entities in the input, it can execute SPARQL queries with EXE to find candidates for multiple entities at once. Multilingual inputs and synonyms can be handled by adding the corresponding data to the search indices underlying GRASP's search functions. For maximum flexibility, annotations can be made individually via ANN, checked via SAN, and removed via DAN at all times during the interaction.

The annotation functions above are tailored to table inputs, because we evaluate our approach on the CEA task (see Section 3). For other input formats, like regular text, the function signatures and implementations would need to be adapted slightly, but the overall idea stays the same. Let us denote the set of all entity linking functions for CEA as CEA = {ANN, DAN, SAN, STP} for later reference.

Importantly, we always set `know_before_use: true`. This is a configuration option of GRASP that restricts the agent to SPARQL queries on items seen during the interaction. See [6] for more details about this option. For entity linking, we extend the scope of this option to also apply to the ANN function, meaning that only knowledge graph items seen during the interaction can be used for annotation. This ensures that all annotations are grounded in knowledge graphs and not hallucinated.

Our implementation supports annotating a subset of rows and/or columns for a given table, varying the number of context rows, as well as providing already known annotations (e.g. useful for incremental row-wise annotation). See Fig. 1 for an exemplary trace produced by our approach for the CEA task.

Table 1
An exemplary table from the MammoTab dataset together with plausible annotations for the cell entity annotation task in parentheses after the cell values. Similar to this example, all tables in the MammoTab dataset have uninformative column headers, which increases the difficulty of the task.

col0	col1	col2
1976 (Q2480)	Eat My Dust! (Q3576864)	Charles Byron Griffith (Q2958461)
1976 (Q2480)	Hollywood Boulevard (Q2709894)	Joe Dante (Q455279)
1976 (Q2480)	Hollywood Boulevard (Q2709894)	Allan Arkush (Q2837668)
1977 (Q2481)	Grand Theft Auto (Q1747227)	Ron Howard (Q103646)

Table 2
Number of indexed knowledge graph items (entities and properties) in millions for our Wikidata dump and the SemTab2025 challenge Wikidata dump 20240720 for all considered languages. The last column refers to the size of the final merged multilingual index respectively.

Knowledge graph	DE	EN	ES	FR	IT	NL	PT	RU	Total
Wikidata	15.1	84.0	14.7	15.9	10.3	55.2	9.2	7.9	89.1
Wikidata 20240720	22.7	88.9	22.5	23.3	18.1	63.1	17.0	7.9	93.6

3. SemTab 2025 challenge - MammoTab track

We evaluate our approach on the CEA task of the MammoTab track of the SemTab 2025 challenge. CEA is the task of assigning to each cell in a given input table the corresponding entity from a given knowledge graph, or *NIL* if there is no such entity. In this challenge, the table inputs come from a subset of an updated version of the MammoTab dataset [7] and have to be linked to the Wikidata knowledge graph [8]. In total, 84,907 table cells from 870 tables have to be annotated with entities. See Table 1 for an exemplary MammoTab table, with annotations. There is no separate training or validation dataset with ground truth annotations, forcing participants to either use data from previous challenges in the series for tuning or, as we do, perform CEA in a zero-shot setting. The CEA task is scored using an F_1-score based on the number of correct cell annotations. Since all cells that need to be annotated are known a priori in this task, the F_1-score is equal to a simple accuracy measure if one produces an annotation for each cell. Multiple submissions are allowed, but only the score of the best solution so far is visible on the challenge website for each participant.

Our approach annotates each row of each table individually. We do this because the tables from MammoTab can get reasonably large, containing up to 256 cells, which is problematic in two ways: first, the annotation process for the whole table might not fit in the context window of the underlying LLM; second, longer annotation processes are more time- and memory-consuming. However, for each row to be annotated, we provide the 10 rows before and after that row as context, without taking the annotations from already annotated rows from the same table into account. In a side experiment, we found that ignoring already existing annotations from other rows did not decrease annotation quality significantly. In fact, note that giving the LLM access to its own annotations for other rows also has the potential to deteriorate the overall result quality by propagating errors.

If the agent tries to annotate a row different from the one to annotate, an error message is returned for the corresponding function call and no annotation made. We use two different sets of functions for the GRASP agent for our evaluations: $\text{CEA}_1 = \text{CEA} \setminus \{\text{SAN}\} \cup \{\text{EXE}, \text{LST}, \text{SPE}, \text{SOP}, \text{SPR}, \text{SEN}\}$, and $\text{CEA}_2 = \text{CEA}_1 \cup \{\text{SAN}, \text{SCN}, \text{SAC}\}$. For our first submission where we used CEA_1, we did not have a dedicated SAN function yet. For the later submissions we added this function as well as GRASP's two advanced functions for constrained search. To support multilingual searches we build multilingual search indices for GRASP from English, German, Spanish, French, Italian, Dutch, Portuguese, and Russian Wikidata labels and aliases. We do this both for a Wikidata dump from October 2024, which we already had set up on our machines, as well as for the official challenge Wikidata dump linked on the SemTab2025 website. See Table 2 for an overview.

Table 3
F_1-scores achieved by our solutions in submission order, together with the used model, average number of steps taken to annotate a row, number of *NIL* annotations, and other details such as the used function set. Each submission improved the leaderboard score of the previous one, which is why we can show exact scores here.

Sub.	F_1-score	Model	Steps	*NIL*	Details
1	72.8	Qwen3 30B A3B Thinking [9]	5.7	17.1k (20.1%)	CEA_1 function set, top k of 10 for `LST` and search functions, `know_before_use: true`
2	75.6	Qwen3 Next 80B A3B [9]	9.6	11.2k (13.2%)	CEA_2 function set, top k of 20 for `LST` and search functions, `know_before_use: true`
3	75.8	Qwen3 Next 80B A3B [9]	9.6	10.7k (12.6%)	CEA_2 function set, top k of 20 for `LST` and search functions, `know_before_use: true`, Wikidata 20240720

Table 4
Number of function calls per function for all submissions, approximately ordered from least to most calls. The first submission used a different set of functions and model compared to the second and third, leading to a very different call distribution.

Sub.	Set	DAN	SAC	SPE	SOP	EXE	SAN	SPR	SCN	LST	STP	ANN	SEN
1	CEA_1	0.1k	-	0.1k	0.2k	1.4k	-	0.3k	-	1.5k	30.2k	75.4k	79.2k
2	CEA_2	1.3k	1.6k	2.9k	2.7k	8.9k	14.2k	14.5k	18.5k	20.7k	36.4k	105.2k	131.4k
3	CEA_2	1.1k	1.8k	2.8k	3.5k	6.2k	12.5k	13.4k	22.5k	21.6k	36.6k	108.3k	129.6k

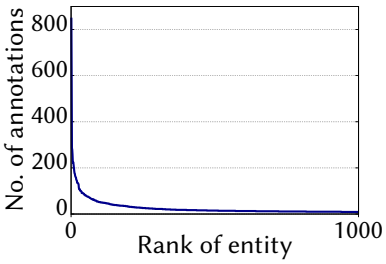
Results

We made three major submissions[1] for the MammoTab track, the results of which are shown in Table 3. All submissions used open-source LLMs from the Qwen3 [9] family. The first one used Qwen3 30B A3B Thinking, a small reasoning model providing a good balance between quality and inference time, together with the CEA_1 function set and GRASP's default top k value of 10 for `LST` and search functions. For the second submission, we used the larger and more recent Qwen3 Next 80B A3B model in its non-reasoning variant, together with the extended function set CEA_2 and a more generous top k value of 20. As expected, the second submission improved the score compared to the first, though only by a rather small margin of 2.8 p.p.

Interestingly, our second submission takes more steps on average to annotate each row, produces significantly fewer *NIL* annotations, and shows a very different call distribution compared to our first submission, performing more searches, `LST` and `EXE` calls. See Table 4 for full details about the last aspect. Considering that each row has on average 3.5 columns to annotate, the model of the first submission only takes 1.6 steps to annotate a column on average, whereas the model of the second submission takes about 2.7. Between the second and third submission, we only changed the Wikidata version from our own to the one recommended by the challenge. However, this only leads to a tiny improvement of 0.2 pp and no significantly different annotation behavior of the agent. In Fig. 2 we show some statistics about the annotations made by our third submission. The curve in this figure shows a Zipf-like distribution, where there are few frequent entities and a long tail of infrequent ones. Overall, 29,743 distinct entities were used to annotate 84,907 table cells, which shows a good diversity in the dataset.

To serve the LLMs we use vLLM [10] and one (Qwen3 30B A3B Thinking) or two (Qwen3 Next 80B A3B) NVIDIA H100 GPUs. All submissions took 3-5 days to finish with 3-6 workers running in parallel. The average time to annotate a single row ranged from 47.2 to 51.6 seconds across all submissions.

[1]There was one other experimental submission, and one accidental duplicate submission.



Entity	#
United States (Q30)	851
Conservative Party (Q9626)	589
Labour Party (Q9630)	286
Liberal Party of Canada (Q138345)	279
2010 United Kingdom general election (Q215622)	258
No. of distinct entities used for annotation	**29,743**

Figure 2: On the left, we show the annotation counts of the 1000 most frequently annotated entities from our third submission. On the right, we show a table with the five most frequently annotated entities, most of which belong to the political domain, as well as the number of distinct entities used for annotation.

4. Conclusion

We present an approach for knowledge graph entity linking that is based on an LLM agent interacting with knowledge graphs by calling functions to search and query them. We evaluate the approach on the SemTab 2025 MammoTab track, where the task is to annotate 84,907 cells from 870 tables with Wikidata entities, and reach first place with a final F_1-score of 75.8%, clearly outperforming other participants.

Currently, our functions for adding, removing, and showing entity links (ANN, DAN, and SAN) are adapted to the tabular inputs of the cell entity annotation task. For future work, we aim to generalize the function interface to support general-purpose entity linking on arbitrary natural-language inputs.

Acknowledgments

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 499552394 – SFB 1597.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] K. Balog, Entity-Oriented Search, volume 39 of *The Information Retrieval Series*, Springer, 2018.
[2] Ö. Sevgili, A. Shelmanov, M. Y. Arkhipov, A. Panchenko, C. Biemann, Neural entity linking: A survey of models based on deep learning, Semantic Web 13 (2022) 527–570.
[3] H. Bast, M. Hertel, N. Prange, A fair and in-depth evaluation of existing end-to-end entity linking systems, in: EMNLP, Association for Computational Linguistics, 2023, pp. 6659–6672.
[4] S. Walter, H. Bast, GRASP: Generic Reasoning And SPARQL Generation across Knowledge Graphs, in: ISWC, 2025. Accepted for publication. Preprint available at https://arxiv.org/abs/2507.08107.
[5] SemTab 2025 Organizers, SemTab 2025 - Semantic Web Challenge on Tabular Data to Knowledge Graph Matching - LLMs & Tabular Data Matching, https://sem-tab-challenge.github.io/2025/, 2025.
[6] S. Walter, H. Bast, GRASP: Generic Reasoning And SPARQL Generation across Knowledge Graphs - Demo System, in: ISWC (Posters, Demos & Industry Tracks), 2025. Accepted for publication.
[7] M. Marzocchi, M. Cremaschi, R. Pozzi, R. Avogadro, M. Palmonari, MammoTab: A giant and comprehensive dataset for semantic table interpretation, in: SemTab@ISWC, volume 3320 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 28–33.
[8] D. Vrandecic, M. Krötzsch, Wikidata: A free collaborative knowledgebase, Commun. ACM 57 (2014) 78–85.
[9] Qwen Team, Qwen3 Technical Report, 2025.

[10] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, I. Stoica, Efficient Memory Management for Large Language Model Serving with PagedAttention, in: Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles, 2023.