Evaluating Language Models on Entity Disambiguation in Tables

Federico Belotti
DISCo, University of Milan-Bicocca
Milan, Italy
federico.belotti@unimib.it

Fabio Dadda
DISCo, University of Milan-Bicocca
Milan, Italy
fabio.dadda@unimib.it

Marco Creamschi
DISCo, University of Milan-Bicocca
Milan, Italy
marco.cremaschi@unimib.it

Roberto Avogadro
SINTEF
Oslo, Norway
roberto.avogadro@sintef.no

Riccardo Pozzi
DISCo, University of Milan-Bicocca
Milan, Italy
riccardo.pozzi@unimib.it

Matteo Palmonari
DISCo, University of Milan-Bicocca
Milan, Italy
matteo.palmonari@unimib.it

Abstract

Tables are crucial containers of information, but understanding their meaning may be challenging. Indeed, recently, there has been a focus on Semantic Table Interpretation (STI), *i.e.*,, the task that involves the semantic annotation of tabular data to disambiguate their meaning. Over the years, there has been a surge in interest in data-driven approaches based on deep learning that have increasingly been combined with heuristic-based approaches. In the last period, the advent of Large Language Models (LLMs) has led to a new category of approaches for table annotation. The interest in this research field, characterised by multiple challenges, has led to a proliferation of approaches employing different techniques. However, these approaches have not been consistently evaluated on a common ground, making evaluation and comparison difficult. This work proposes an extensive evaluation of four state-of-the-art (SOTA) approaches — Alligator (formerly s-elBat), Dagobah, TURL, and TableLlama; the first two belong to the family of heuristic-based algorithms, while the others are respectively encoder-only and decoder-only LLMs. The primary objective is to measure the ability of these approaches to solve the entity disambiguation task, with the ultimate aim of charting new research paths in the field.

CCS Concepts

• **Computing methodologies → Machine learning**; • **Information systems → Ontologies**.

Keywords

Semantic Web, Knowledge Base, Knowledge Base Construction, Knowledge Base Extension, Knowledge Graph, Semantic Table Interpretation, Table Annotation, Data Enrichment, Tabular Data

1 Introduction

Tables are commonly used to create, organise, and share information in various knowledge-intensive processes and applications in business and science. Disambiguating values occurring in the table cells using a background Knowledge Graph (KG) is useful in different applications. First, it is part of the broader objective of letting machines understand the table content, which has been addressed by different research communities with slightly different formulations like Semantic Table Interpretation (STI) [49] - the one considered in this paper, semantic labelling [62], and table annotation [68]. The main idea behind these efforts is to match the table against a background knowledge graph by annotating cells (mentions) with entities (Cell-Entity Annotation - CEA), columns with class labels (Column-Type Annotation - CTA), and pairs of columns with properties (Column-Property Annotation - CPA) [37]. Second, annotations produced by table-to-graph matching algorithms can be used to transform the tables into Knowledge Graphs (KGs) or populate existing ones. Third, links from cells to entities of KGs support data enrichment processes by serving as bridges to augment the table content with additional information [20]. This conceptualisation covers most of the proposed definitions by generalizing some aspects (*e.g.*, consideration of NILs and selection of column pairs to annotate).

In particular, we focus on Entity Linking (EL) in tables (CEA, in the STI terminology), which is relevant not only to support table understanding but also to support data transformation, integration and enrichment processes, which are particularly interesting from a data management point of view. The Cell-Entity Annotation (CEA) tasks can be broken down into two sub-tasks: candidate Entity Retrieval (ER), where a set of candidates for each mention is collected and, often, associated with an initial score and rank, and Entity Disambiguation (ED), where the best candidate is selected (and, in some case, a decision whether to link or not is also considered [6]).

When considering approaches to STI and, especially, CEA, it should be considered that the content and the structure of tables may differ significantly, also depending on application-specific features: column headers may have interpretable labels or be omitted; the number of rows can vary from a dozen (*e.g.*, as typical in tables published on the web or in scientific papers) to hundreds thousand or even millions (*e.g.*, in business data); the cells may include reference to well-known entities (*e.g.*, geographical entities) as well as to specific ones (*e.g.*, biological taxa); tables may come with a rich textual context (*e.g.*, caption or other descriptions in web or scientific documents), or no context at all (*e.g.*, in business data) [49].

A first generation of approaches to CEA has exploited different matching heuristics, traditional machine learning approaches based on engineered features (in the following “feature-based ML”), or a combination of both [49]. The International Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab), at its sixth edition in 2024, is a community-driven effort to compare different approaches systematically with a common experimental setting. The SemTab challenge has pushed different researchers to increase the performance of their approaches and publish datasets for evaluating STI approaches. For example, some effort has been dedicated to optimising entity retrieval, considering the limitations of approaches based on SPARQL queries or Wikidata lookup services [8]. A few methods from this research community have included embeddings based on LLMs and graphs to support some tasks [22, 32], including CEA, yet at a limited scale.

The increased recognition of the power of LLMs has led to a new generation of Generalistic Table Understanding and Manipulation (GTUM) approaches that support STI (CEA, CTA and CPA) and other tasks (*e.g.*, question answering, schema augmentation, row population and fact verification among the others). The first remarkable example is TURL [23], which is based on an adaptation of BERT [24] to consider the tabular structure, additionally fine-tuned to execute specific tasks (including ED). The latest of these models, TableLlama [80], is based on the autoregressive Large Language Model (LLM) Llama 2-7B [72], which is fine-tuned with instruction tuning to perform specific tasks. The approach exploits the power of instruction tuning with large contexts to convert ED (and other tasks) into a prompt-based text generation task. These approaches have been trained and tested on datasets that fit their generalistic ambition and reported as SOTA approaches to the considered tasks.

While STI-related approaches and GTUM approaches support ED, an evaluation of these families of approaches on a common experimental ground is missing. In addition, the evaluation of GTUM approaches has focused on their performance on different tasks rather than drilling down on CEA; as a result, it is unclear to what extent these approaches reach SOTA performance under different settings (*e.g.*, also considering different types of tables). Finally, several aspects of CEA that are important from a data management point of view, *e.g.*, scalability, have not been considered at all. We posit that several of these questions are important to understand the current status of ED algorithms for tables, as well as the challenges to be addressed in the future.

With our study, we want to provide a more detailed analysis of the advantages and disadvantages of GTUM ED approaches based on LLMs, especially when compared to those specifically focusing on CEA developed in the context of STI.

Our analysis mainly considers the SOTA generalistic generative model TableLlama [80], its predecessor TURL [23], a BERT-based encoder-only transformer, and Alligator [6], a recent feature-based Machine Learning (ML) approach. These models represent three different inference mechanisms for ED, sketched in Figure 1, each one associated with a set of expected advantages: TableLlama is expected to exploit implicit knowledge of a large LLM; TURL is expected to be a more efficient generalistic model based on a small LLM fine-tuned specifically for the ED task; Alligator exploits a set of features engineered based on the experience with the SemTab challenge, which is further processed by two neural networks to return confidence scores. Each approach is tested in *in-domain*, *out-of-domain*, and *moderately out-of-domain* settings, as more precisely defined in Section 4.2. In addition, we test the moderately out-of-domain fine-tuning of LLMs-based approaches to test generalisation and adaptation capabilities. Finally, we provide some results with an approach that achieved top performance in previous SemTab challenges[1].

We remark that this paper aims to provide insights on the behavior and impact of GTUM models on ED rather than introducing a new approach. This is inspired by the large body of similar analyses addressing specific problems in NLP [39, 40, 74, 81, 84]. In our work, we also develop an evaluation protocol that can be used in future work.

To summarise, our **main contributions** are:

(1) Test the performance of different genres of STI models when used in combination with a realistic candidate retrieval step, performed by the engineered tool *LamAPI* [8];
(2) Test the performance of these models on several datasets used to evaluate SOTA STI approaches, through a common-ground comparison with approaches trained on these data;
(3) Assess the performance improvement by adaptation with additional moderately-out-of-domain fine-tuning (Sec. 4.2);
(4) Evaluate the computational efficiency and provide implications for actual usage in different application settings.

This paper is organised as follows: Section 2 proposes a detailed examination of the techniques used by STI approaches in the SOTA. Section 3 introduces and details the approaches tested in this work, relating them to the ED challenges they are intended to solve. Section 4 describes the objectives of this study, the datasets used to evaluate the selected approaches and defines the experimental settings followed in Section 5, which introduces the configuration parameters and the evaluation results, discussing the main results and the ablation studies. Finally, we conclude this paper and discuss the future direction in Section 6.

2 Related Work

CEA is usually divided into two sub-tasks: *retrieval of candidate entities*, *i.e.*, ER [64], and *entity disambiguation*, *i.e.*, ED [63], where one or no candidate is selected as a link, usually after scoring and ranking the candidates. Approaches to ED for tables have been mostly proposed as part of broader STI systems [49]. For this paper, we group these approaches into two main categories: i) those based on heuristics, (feature-based) ML and probabilistic approaches, and ii) those based on LLM.

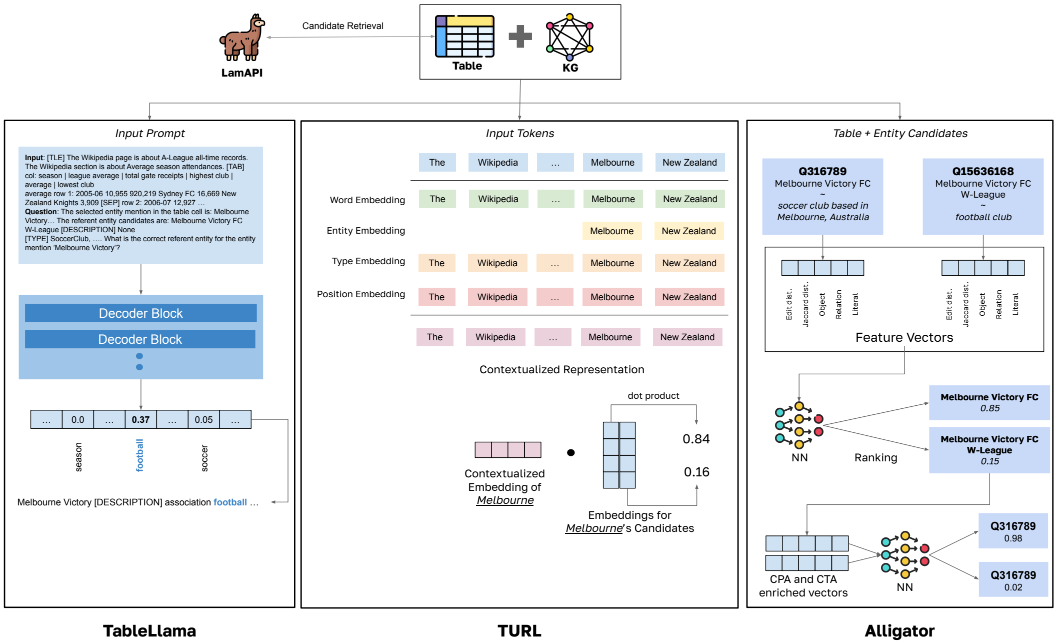
Heuristic, ML and Probabilistic-based Approaches. We refer to [49] for a thorough review of STI approaches and ED approaches proposed therein up to 2022. The ED task in the STI can be performed by applying multiple techniques while focusing on different inherent information: i) *similarity*, ii) *contextual information*, iii) *ML techniques*, and iv) *probabilistic models*. Often, the disambiguation step involves selecting the winning candidate based on **heuristics**, like the string *similarity* between the entity label and mention [3, 7, 14, 17–20, 33, 34, 44, 48, 59, 66, 67, 71, 73, 82].

Contextual information during the CEA task considers the surrounding context of a table cell, such as neighbouring cells, column headers, or header row. Contextual information provides additional clues or hints about the meaning and intent of the mention. By analysing the context, a system can better understand the semantics of the cell and make more accurate annotations [2, 7, 9, 10, 12–14, 17, 19, 25, 32–34, 50, 51, 56–58, 65, 70, 71].

Other methods that can be employed are **ML techniques**. These techniques typically involve training a ML model on a labelled

[1] With some limitations due to its excessive execution time and the difficulty of properly replicating the published results despite our best effort.

Figure 1: Architectures of TableLlama, TURL and Alligator.



Input: [TLE] The Wikipedia page is about A-League all-time records. The Wikipedia section is about Average season attendances. [TAB] col: season | league average | total gate receipts | highest club | average | lowest club average row 1: 2005-06 10,955 920,219 Sydney FC 16,669 New Zealand Knights 3,909 [SEP] row 2: 2006-07 12,927 …
Question: The selected entity mention in the table cell is: Melbourne Victory… The referent entity candidates are: Melbourne Victory FC W-League [DESCRIPTION] None
[TYPE] SoccerClub, …. What is the correct referent entity for the entity mention 'Melbourne Victory'?

Melbourne Victory [DESCRIPTION] association football …

dataset where cells are annotated with their corresponding entities. The models, like Support Vector Machine (SVM) [53], Neural Network (NN) [71] and Random Forest [79], learn patterns and relationships between the cells content and their associated entities. To predict the most appropriate entity, ML techniques consider various cell features, such as the textual content, context, neighbouring cells, and other relevant information. Alligator is a recent approach belonging to this family [6]; it is used in this study and further described in Section 3.

Probabilistic models are frameworks for representing and reasoning under uncertainty using probability theory. These models vary in their representation of dependencies and use diverse graphical structures. Several Probabilistic Graphical Model (PGM) can also be used to resolve the disambiguation task, such as Markov models or Loopy Belief Propagation (LBP) [11, 44, 52, 54, 78].

LLMs-based approaches. In the current SOTA, several attempts to apply **LLM** in the STI process can be identified. Based on the architecture structure of LLMs, approaches can be categorised into two groups: i) *encoder-based*, and ii) *decoder-based* [60].

Starting from *encoder-based approaches*, Ditto [47] utilizes Transformer-based language models to perform a slightly different task; in fact, the goal is entity-matching between different tables. Doduo [68] performs Column-Type Annotation (CTA) and CPA using a pre-trained language model, precisely fine-tuning a BERT model on serialised tabular data. Column (columns-properties) types are predicted using a dense layer followed by an output layer with a size corresponding to the number of column (columns-properties) types. Dagobah SL 2022 [32] employs an ELECTRA-based [16] cross-encoder, a variant of the BERT model, which takes a concatenated input, including table headers and the entity description, and outputs a probability value representing the entity's likelihood concerning the headers. TorchicTab [22] is composed of two subsystems: TorchicTab-Heuristic and TorchicTab-Classification, with the latter that utilises Doduo [68] internally.

Regarding the *decoder-only approaches*, [42] explored the CTA task by employing ChatGPT, performing experiments with diverse prompts tailored for the task using a subset of the SOTAB benchmark [43]. Another study evaluates GPT3.5-turbo-0301 in zero-shot settings on a task that was somehow related to CEA; the task consisted of classifying descriptions of products based on attribute-value pairs [61]. Some works included structured tabular data into the training process of general purpose decoder-based LLMs to address the peculiarities of specific domains, *e.g.*, BloombergGPT [77] for the financial domain or CancerGPT for the medical one [46]. Other works, instead, focused on table-specific tasks, especially reasoning [41, 55, 69], enhanced by Chain-of-Thought (CoT) [75, 83]. Another decoder-based approach, TableGPT [45], performs several tasks, including CTA using a fine-tuned version of GPT-3.5.

Of particular interest for our work, LLM-based models capable of addressing all the STI tasks and in particular the CEA one are: i) TURL [23], which leverages a pre-trained TinyBERT [36] model to initialise a structure-aware Transformer encoder, fine-tuned to obtain contextualised representations for each cell, with matching scores between the KG candidates' representations and cell embeddings that are calculated using a linear function, and transformed into a probability distribution over the candidate entities; ii) TableLlama [80], a fine-tuned Llama2 on a multi-task dataset for tabular data, performs CEA (along with several other tasks), where the entity linking sub-dataset derives from the TURL [23] dataset, by prompting the LLM to retrieve the correct candidate given the serialized table, the table metadata, the cell to be linked and a set of proposed candidates.

A more comprehensive review of both *encoder-only* and *decoder-only* LLM-based approaches for Table Understanding can be found in [27].

3 Considered Approaches

We select four approaches representative of different categories of algorithms that solve semantic tasks on tables, and, especially CEA: TableLlama is the first autoregressive LLM that is specifically instruction-tuned on tabular data and reports SOTA results [80]; TURL, cited as previous SOTA in [80], is a BERT-based encoder-only model performing the three STI tasks, including CEA [23]; Alligator is a recent feature-based ML algorithm focusing on CEA, is publicly available, and has been evaluated in settings similar to the moderately-out-of-domain settings discussed in this paper [6]; Dagobah, a heuristic-based algorithm, has been the winner of various rounds of the SemTab challenge in 2020 [34], 2021 [33] and 2022 [32] and is publicly available.

We remark that all the selected approaches perform all three STI tasks (which is rare and not obvious, especially for the LLM-based approaches), *i.e.*, CTA, CPA and CEA, and we focus only on the CEA one.

TURL [23] (Figure 1) introduces the standard BERT pre-training finetuning paradigm for relational Web tables and is composed of three main modules: i) an embedding layer to convert different components of a table into embeddings, ii) a pre-trained TinyBERT Transformer [36] with structure-awareness to capture both the textual information and relational knowledge, and iii) a final projection layer for pre-training/fine-tuning objectives. The embedding layer is responsible for embedding the entire table, distinguishing between word embeddings $\mathbf{x}_w$, derived from table metadata and cell text (mentions), and entity embeddings $\mathbf{x}_e$, representing the unique entities to be linked. The sequence of tokens in $\mathbf{x}_w$ and the entity representation $\mathbf{x}_e$ are sent to a structure-aware TinyBERT Transformer to obtain a contextualised representation. To inject the table structural information into the contextualised representations, a so-called visibility-matrix M is created, such that entities and text content in the same row or column are visible to each other, except table metadata that are visible to all table components. During pre-training, both the standard *Masked Language Model* (MLM) and the newly introduced *Masked Entity Retrieval* (MER) objectives are employed, with the projection layer that is learned to retrieve both the masked tokens and entities[2]. During fine-tuning, the model is specialised to address the specific downstream task. Specifically, for CEA, when provided with the sub-table containing all mentions to be linked to an external KG, TURL is fine-tuned to produce a probability distribution over the combined set of candidates for each mention to be linked. This means that TURL lacks the context provided by all the not-to-be-linked cells, without the possibility of abstaining from answering or responding with a NIL. Also, in the original paper, we observe that the best candidate is chosen by a function that compares the best score computed by TURL and the score assigned by the model to the first candidate retrieved by the Wikidata-Lookup service by down-weighting the best model prediction. As in previous comparisons [80], to evaluate TURL's performance on the ED task, we consider the model predictions only. Finally, as TURL accepts the whole table as input, computational performance varies depending on the size of the input table.

TableLlama [80] (Figure 1) employs Llama2 [72] and instruction tuning [76] to solve multiple tasks related to tables, *e.g.*, CTA, CPA, CEA and Q&A to name a few. To this end a multi-task instruction tuning dataset named TableInstruct is made available, containing more than 1.24M training tables and 2.6M instances gathered from 14 different table datasets of 11 distinctive tasks. To account for longer tables, exceeding the maximum context length of Llama2 (which is fixed at 4096 tokens), LongLora [15] is used to enlarge the maximum context length to 8192 tokens. In particular, the LLM is prompted with i) an instruction that describes the high-level task to be solved, ii) an input prepended by the [TLE] special character followed by the table metadata, if available, and the serialised table and iii) a question based on the task to be solved by the LLM. In particular, the table starts with the [TAB] special character and is followed first by the table header and then by every row in the table, separated by the special character separator [SEP]. For the CEA task, the question asks the LLM to link a particular mention found in the table against a small set of candidates (maximum 50 candidates) by extracting the correct candidate from the proposed list without, as in TURL, the possibility of not answer or to answer with a NIL. Since the context is fixed to 8192 tokens, compromises must be made to reduce either the length of the set of candidates or the input table.

Alligator (Figure 1) is a feature-based ML approach based on multiple steps: i) *data analysis and pre-processing*, ii) *ER*, iii) *local feature extraction*, iv) *local scoring*, v) *feature enrichment*, vi) *context-based scoring*. The *data analysis and pre-processing* step converts all cells to lowercase and removes extra spaces and special characters, e.g., underscores (_), to improve the results of the entity retrieval phase. In addition, columns are classified as either *L-column* (containing literals) or *NE-column* (containing named-entity mentions). The ER step extracts relevant candidates from the KG using the *LamAPI* services. The *local feature extraction* step builds a vector of engineered features for each candidate entity; the vector represents information about the specific candidate (e.g., popularity, number of tokens) or its comparison with the values in the cell and the row (different similarity scores, matches with other cells on the row, matches with the entity description, and so on). *Local scoring* computes a matching score for each candidate, using a simple deep Neural Network (NN): the NN takes the local feature vectors as input and is trained to convert the features in a normalised matching score. In practice, this step re-ranks all the candidates, considering mainly text similarity and matches against values on the same *row*. The last steps compute updated scores for the candidate of a given cell by considering the best candidates for other cells on the same *column*. In practice, Columns-Property Annotation (CPA) and CTA tasks are executed on a best-effort basis to capture, for each candidate, the degree of agreement between its types and properties and the types of properties of other best candidates in the same column. After this *feature enrichment* step, a simple deep NN similar to the

[2] The entities are retrieved from a small candidates set, considering that entity vocabulary could be quite large

Table 1: Statistics of the datasets used to pre-train the considered approaches

Approach	Dataset	Entities	Source	Entity Domain
Alligator	SemTab2021 - R2	47.4K	Graph Queries	Cross
	SemTab2021 - R3	58.9K	Graph Queries	
	SemTab2022 - R2 (2T)	994.9K	Graph Queries / Web Tables	
	SemTab2020 - R4	667.2K	Graph Queries	
	SemTab2019 - R3	390.4K	Graph Queries	
	SemTab2019 - R1 (T2D)	8K	Graph Queries / Web Tables	
	Total	2.16M		
TURL [23]	TURL WikiTable w/o WikiGS [26]	1.23M	Web Tables	Cross
TableLlama [80]	TableInstruct [80]	2.6M	Web Tables	Cross

previous one predicts the updated normalised scores for all the candidates[3].

Dagobah [33] performs the CEA, CPA and CTA tasks by implementing a multi-step pipeline: i) table pre-processing, ii) ER, iii) candidate pre-scoring, iv) CPA, CTA, and v) CEA. The *table pre-processing* step involves generating metadata for the table, including orientation detection, header detection, key column detection, and column primitive typing. These tasks help identify the structure and annotation targets, facilitating subsequent annotation steps. ER retrieves relevant candidate entities from the KG for each (entity) cell in the table using Elasticsearch; it considers the primitive types identified during pre-processing and enriches entity information with aliases to increase coverage. *Candidate pre-scoring* computes a relevance score for each candidate by combining two features: context and literal similarity. Given a cell to disambiguate and a candidate entity, the context feature compares the cell neighbours to the candidate neighbours using a smart procedure. Overall, this method improves the precision of context scoring by incorporating both direct and indirect connections while avoiding overly complex and noisy paths. Then, CPA and CTA modules identify the most suitable relations and types for column pairs and target columns using a majority voting strategy. Finally, the most relevant entity for each table cell is selected by combining pre-scoring with information from CPA and CTA into a final score. The main challenges with Dagobah are its high time complexity and memory usage, making it impractical for processing large tables. This difficulty stems from the expensive task of matching table rows with relations in a KG. Consequently, we limited our evaluation of DAGOBAH to the datasets *HT-R1*, *HT-R2*, and *WikidataTablesR1* because these datasets contain tables with limited size, as detailed in Section 4.1.

Table 1 lists the datasets used to train the selected approaches; in particular, for TableLlama we use the published model on HugginFace, while we train TURL and Alligator on the same datasets used in the original paper. We use the term "pre-train" to refer to this *main* training phase to distinguish it from subsequent fine-tuning (see Section 4.4). Dagobah is not listed because is not based on training (we use the open-source code).

4 Study Set-up

As specified by the SemTab challenge [30], the metrics adopted to evaluate an approach are: Precision (P) = $\frac{\#\ correct\ annotations}{\#\ submitted\ annotations}$, Recall (R) = $\frac{\#\ correct\ annotations}{\#\ ground-truth\ annotations}$ and F1 = $\frac{2PR}{P+R}$. Since we are mainly interested in measuring the ability of the models to disambiguate the correct entity among a limited set of relevant candidates, **we ensure the correct entity is always present in the candidates set, and otherwise, we inject it**, as in previous work [23, 80]. Under this setting, Precision, Recall and F1 are the same and equal to the Accuracy metric. Therefore, we report the accuracy as $\frac{\#\ correct\ annotations}{\#\ mentions\ to\ annotate}$.

To maintain a good balance between speed, coverage, and diversity, we retrieve and inject at most 50 candidates for every mention, eventually adding the correct one.

Our evaluation has the main objective of evaluating the selected approaches on datasets not considered in the original experiments, considering especially the evaluation of TURL and TableLlama on STI-derived datasets and of Alligator and Dagobah (with some limitations) on the datasets used to train and evaluate the first two approaches. Therefore, we first discuss the datasets used in our study (Section 4.1), and the evaluation protocol with its associated objectives (Section 4.1). Then, we provide details about ER (Section 4.3) and the training of the models (Section 4.4).

4.1 Datasets

The datasets considered in this work come from different sources and contain information about domains. In particular, we have selected the following:

- **SemTab2021 - R3 (BioDiv)** [4]: the BioDiv dataset is a domain- specific benchmark comprising 50 tables from biodiversity research extracted from original tabular data sources; annotations have been manually curated.
- **SemTab2022 - R2 (2T)** [21]: the dataset consists of a set of high-quality manually-curated tables with non-obviously linkable cells, *i.e.*, where mentions are ambiguous names, typos, and misspelt entity names;
- **SemTab2022 - R1 & R2 (HardTables)** [1]: datasets with tables generated using SPARQL queries [37]. The datasets used from HardTables 2022 are *round 1* (R1) and *round 2* (R2). The target KG for this dataset was Wikidata, and as with previous years, the tasks were CEA, CTA, and CPA;
- **SemTab2023 - R1 (WikidataTables)** [30]: datasets with tables generated using SPARQL queries for creating realistic-looking tables. The dataset includes *Test* and *Validation* tables, yet we exclusively employ the *Validation* tables due to Gold Standard (GS) being provided. The target KG for

[3] In principle, the approach also estimates a confidence score for each cell to make a decision whether to link the best candidate or not; however, in this paper, we focus on the ED task and consider the candidate with the best score as the output of Alligator

Table 2: Statistics of the datasets used to fine-tune and evaluate models. †indicates datasets that have been sub-sampled so that they can be entirely processed by TableLlama within its 8192 context. {Dataset}-red means that the specific dataset has been reduced as explained in Section 4.1.

Gold Standard	Dataset	Split	Tables	Cols min | max | $\bar{x}$	Rows min | max | $\bar{x}$	Entities
SemTab2021	Biodiv-**red**	Test†	11	1 | 26 | 17,45	26 | 100 | 58,90	1 232
SemTab2022	2T-**red**	Train†	91	1 | 8 | 4,86	5 | 369 | 98,61	14 674
		Test†	26	1 | 8 | 4,65	7 | 264 | 74,58	4 691
	R1 (HardTables)	Train	3 691	2 | 5 | 2,56	4 | 8 | 5,68	26 189
		Test	200	2 | 5 | 2,59	4 | 8 | 5,74	1 406
	R2 (HardTables)	Train†	4 344	2 | 5 | 2,56	4 | 8 | 5,57	20 407
		Test	426	2 | 5 | 2,53	4 | 8 | 5,56	1 829
SemTab2023	R1 (WikidataTables)	Train	9 917	2 | 4 | 2,51	3 | 11 | 5,65	64 542
		Test	500	2 | 4 | 2,46	3 | 11 | 6,95	4 247
TURL	120k	Train	13 061	1 | 43 | 5,43	1 | 624 | 12,98	120 000
	2k-**red**	Test	1 295	1 | 14 | 1,03	6 | 257 | 32,95	1 801

this dataset was Wikidata, and the tasks were CEA, CTA, and CPA;

- **TURL** [23]: the authors of TURL developed the TURL dataset (Wikitables) using the extensive WikiTable corpus, a rich compilation of tables from Wikipedia. This dataset includes table metadata such as the table name, caption, and column headers. This dataset has been sub-sampled by TableLlama's authors to create a smaller version containing exactly 2K mentions, called *TURL-2K*, and used as the test set.

Table 2 reports the statistics of each dataset used for testing (and fine-tuning) (†indicates the datasets that have undergone sub-sampling). The SemTab datasets were already split in train and test except for *BioDiv 2021*, which has been only used during the testing phase along with the *TURL-2K*. To create a unified and common experimental setting, each dataset has been modified to contain the same set of mentions: datasets for TableLlama were first created by generating prompts for each mention while filtering out all prompts that exceeded TableLlama's context length (which is equal to 8192 tokens). For this reason, we have renamed *BioDiv*, *2T* and *TURL-2K* into *BioDiv-red*, *2T-red* and *TURL-2K-red.* To reduce the number of tokens generated for each prompt, we have taken some precautions: i) the *description* separator has been reduced from *[DESCRIPTION]* to *[DESC]*, and ii) the row separator (*[SEP]*) has been deleted. Then, TURL datasets and the ground truths for Alligator and Dagobah were created using the same mentions as TableLlama.

4.2 Distribution-aware Experimental Objectives

The datasets used for training or evaluating ED are generated from different sources, contain tables of different sizes, and hold information about disparate domains. We assume each dataset is associated with a data distribution generating it [4, 21, 23, 30, 37, 38].

We introduce the concepts of "*in-domain*", "*out-of-domain*" and "*moderately-out-of-domain*" settings related to the evaluation of a particular approach on a test set, specialising a distinction between "*in-domain*" and "*out-of-domain*" used in [80]. These denominations depend on the source the data has been generated from and the domain(s) it holds the information about. In particular, we define:

- "*in-domain*" (IN): a test set Y for an approach A is considered "*in-domain*" for A if A has been trained on a dataset X generated from the *same* data source and covering *similar* domain(s) as Y;
- "*out-of-domain*" (OOD): a test set Y for an approach A is considered "*out-of-domain*" if A has been trained on a set of data X generated from a *different* data source and covering *different* domain(s) as Y;
- "*moderately-out-of-domain*" (MOOD): a test set Y for an approach A is considered "*moderately-out-of-domain*" for A if A has been trained on a set of data X generated from the *same* data source but covering *different* domain(s) as Y or vice versa, i.e., if A has been trained on a set of data X generated from a *different* source but covering *similar* domain(s) as X. This covers a setting such as the evaluation of TableLlama, pre-trained on the **TURL** dataset [23], on the STI-derived test set **2T**: the two datasets contain different tables but cover cross-domain information linked to Wikidata in a quite similar way.

Given the definitions above, the evaluation procedure has been divided into two steps

1. We assess the performance of the considered approaches on the test data defined in Section 4.1, without further fine-tuning. The heterogeneity of the test data implies all the algorithms are tested against "*in-domain*", "*moderately--out-of-domain*" and "*out-of-domain*" data, as visible in Table 3
2. We fine-tune TURL, TableLlama and Alligator on the train splits from their MOOD data, *i.e.*, on SemTab2022 R1 (HardTables), SemTab2022 R2 (HardTables), SemTab2023 R1 (WikidataTables) and 2T-red for TURL and TableLlama, and on TURL120k[4] for Alligator; finally we test the fine-tuned models on our test data

[4] TURL-120k has been created starting from the original TURL dataset [23] by sub-sampling 13'061 tables containing 120'000 mentions (≈ 125'000 is the number of mentions used to fine-tune TableLlama and TURL on their MOOD data combined).

Table 3: Characteristics of the datasets used to test the approaches based on ML. For every dataset we report its source, the domain of the entities contained, and the characterization in In-Domain (IN), Out-Of-Domain (OOD) and Moderately-Out-Of-Domain (MOOD) for each of the pre-trained models (see Table 1 for the pre-training datasets). {Dataset}-red means that the specific dataset has been reduced as explained in Section 4.1.

Gold Standard	Dataset	Source	Entity Domain	Approaches		
				Alligator	TURL	TableLama
SemTab2021	Biodiv-**red**	Biodiversity Tables	Specific	OOD	OOD	OOD
SemTab2022	2T-**red**	Graph Queries / Web Tables	Cross	IN	MOOD	MOOD
SemTab2022	R1 (HardTables)	Graph Queries	Cross	IN	MOOD	MOOD
SemTab2022	R2 (HardTables)	Graph Queries	Cross	IN	MOOD	MOOD
SemTab2023	R1 (WikidataTables)	Graph Queries	Cross	IN	MOOD	MOOD
TURL	2K-**red**	Web Tables	Cross	MOOD	IN	IN
TURL	120k	Web Tables	Cross	MOOD	IN	IN

The **main objectives** of our analysis can be summarised as follows:

- test the capability of pre-trained approaches on tables from different datasets, which we expect to be representative of different data distributions;
- test the generalisation capability of both feature-based ML and LLM-based approaches after fine-tuning in MOOD settings;
- compare the approaches in inference time and occupied memory to get insights about their applicability on application domains where processing of large tables may be required (e.g., enrichment of business data).
- identify strengths and weaknesses of LLM-based approaches on the ED task with ablation studies.

4.3 Candidate entity retrieval with LamAPI

The candidates for the mentions contained in the TURL dataset [23], along with its sub-sampled version TURL-2K [80], were retrieved through the Wikidata-Lookup-Service[5], which is known to have a low coverage w.r.t. other ERs [8]. For this reason, we researched to identify a state-of-the-art approach/tool specific to the ER. The final choice fell on *LamAPI*, an ER system developed to query and filter entities in a KG by applying complex string-matching algorithms. As suggested in the paper [8], we have integrated DBpedia (v. 2016-10 and v. 2022.03.01) and Wikidata (v. 20220708), which are the most popular KGs also adopted in the SemTab challenge[6]. In *LamAPI*, an ElasticSearch[7] index has been constructed, leveraging an engine designed to search and analyse extensive data volumes in nearly real-time swiftly. These customised local copies of the KGs are then used to create endpoints to provide ER services. The advantage is that these services can work on partitions of the original KGs to improve performance by saving time and using fewer resources. This simulates an application setting of large-scale entity disambiguation (large tables), where a local copy can speed up operations substantially. The *LamAPI Lookup* service was used to extract the candidates, as carried out by other services [6]. Given a string input, the service retrieves a set of candidate entities from the reference KG.

The choice to use *LamAPI* as an ER system is based on its availability and performance compared to other available systems (*e.g.*, Wikidata Lookup) [8]. To further validate the choice of *LamAPI* we have computed the number of mentions with K candidates for the TURL-2K-red dataset: the original TURL - 2K dataset has almost 600 mentions with 1 candidate (the correct one, with 969 mentions (≈ 48%) with at most 5 candidates included the correct one). For all those mentions the Wikidata-Lookup service fails to retrieve something meaningful. On the contrary *LamAPI* retrieves for 1650 mentions (≈ 91%) in our sub-sampled dataset at least 45 candidates. In particular, for the TURL-2K-red dataset, the coverage (*i.e.*, how many times the correct candidate is retrieved by the ER system over the total number of mentions to cover) of *LamAPI* and Wikidata-Lookup is 88.17% and 71.75% respectively. For all these reasons we decided to replace the candidates extracted from *LamAPI* to build a new version of TURL-2K-red dataset which we called *TURL-2K-red-LamAPI*.

4.4 Training and implementation details

Pre-training and model usage. For TURL we first replicated the pre-training with the same hyperparameters as specified by the authors in [23] but in a distributed setting on 4 80GB-A100 GPUs, following the findings in [29], then we fine-tuned it with the default hyper-parameters, matching the CEA results in the original paper. We use the open-source version of TableLlama made available on HuggingFace[8]. Alligator is pre-trained on different SemTab datasets before 2022 as in the original paper[9]. We refer to Table 3 for details about the datasets used for pre-train. Dagobah is run with the default hyperparameters as specified in the corresponding GitHub repository[10].

Fine-tuning. We remind that we consider two evaluation settings (see Section 4.2): 1) the approaches based on their pre-trained state without having directly seen any table of the test datasets; 2) fine-tuning TURL, TableLlama and Alligator on MOOD data (see Table 1). For the fine-tuning of TURL the default hyperparameters have been

[5] https://www.wikidata.org/w/api.php?action=wbsearchentities&search=Obama&language=en&limit=50&format=json, which retrieves at most 50 candidates for the mention "Obama"

[6] www.cs.ox.ac.uk/isg/challenges/sem-tab

[7] www.elastic.co

[8] https://huggingface.co/osunlp/TableLlama

[9] As reported in Table 3, the 2T dataset is one of the datasets used to pre-train Alligator. Even though we have 2T in our test data, those two datasets come from different rounds and years of the SemTab challenge.

[10] https://github.com/Orange-OpenSource/Table-Annotation

Table 4: Performances of the four algorithms on our test data as "Accuracy (Data Typology)", where "Data Typology" represent whether the test set Y is considered IN, MOOD or OOD for the the approach A. Given the heterogeneity of our test data, all the models are tested on different typologies of dataset: "*in-domain* (IN)", "*moderately-out-of-domain* (MOOD)" and "*out-of-domain* (OOD)". MOOD → IN indicates that a model has been fine-tuned on MOOD data. TURL and TableLlama are fine-tuned on their MOOD data, *i.e.*, on the train split of SemTab2022 R1 (HardTables), SemTab2022 R2 (HardTables), SemTab2023 R1 (WikidataTables) and 2T-red (≈ 125k mentions). Alligator is fine-tuned on its MOOD dataset, *i.e.*, TURL-120k (120k mentions). {Dataset}-red means that the specific dataset has been reduced as explained in Section 4.1. "o.o.m" stands for out-of-memory.

Dataset	Pre-trained				Fine-tuned		
	TURL	TableLlama	Alligator	Dagobah	TURL	TableLlama	Alligator
2T-red	0,1343 (MOOD)	0,8243 (MOOD)	0,7156 (IN)	/ (IN)	0,3323 (MOOD→IN)	0,8399 (MOOD→IN)	**0.8531 (IN)**
HT-R1	0,3997 (MOOD)	0,7873 (MOOD)	**0,8890 (IN)**	0,7413 (IN)	0,7454 (MOOD→IN)	0,8001 (MOOD→IN)	0.8165 (IN)
HT-R2	0,2763 (MOOD)	0,6619 (MOOD)	**0,8218 (IN)**	0,6289 (IN)	0,6018 (MOOD→IN)	0,6778 (MOOD→IN)	0.7212 (IN)
WikidataTables-R1	0,3391 (MOOD)	0,7426 (MOOD)	**0,8248 (IN)**	0,7279 (IN)	0,7061 (MOOD→IN)	0,7530 (MOOD→IN)	0.8086 (IN)
BioDiv-red	0,8109 (OOD)	0,9513 (OOD)	0,5674 (OOD)	/ (OOD)	0,6347 (OOD)	**0,9610 (OOD)**	0.8547 (OOD)
TURL-2K-red-LamAPI	0,7118 (IN)	**0,9051 (IN)**	0,6017 (MOOD)	o.o.m (MOOD)	0,5347 (IN)	0,9045 (IN)	0,7456 (MOOD → IN)

Figure 2: Overall elapsed time per dataset.

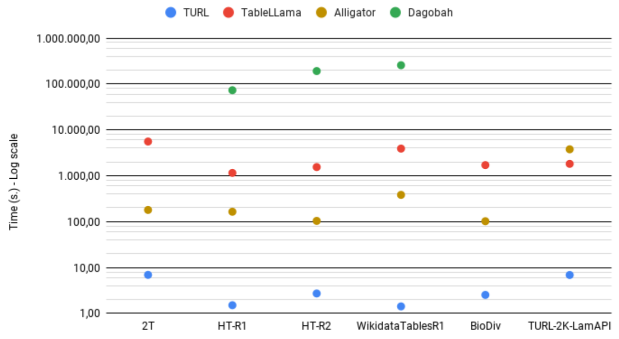
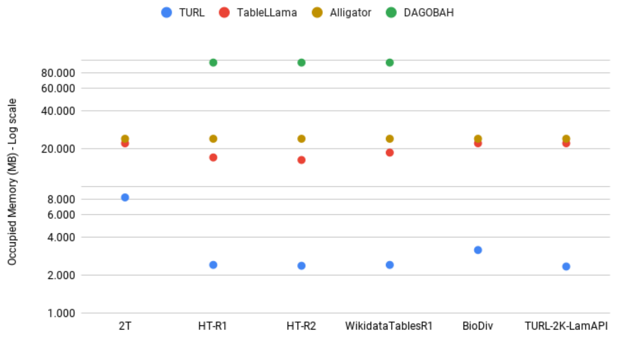


Figure 3: Overall occupied memory per dataset.



adopted as specified by the authors in [23]. For TableLlama we have employed the findings in [35], *i.e.*, rewarming the learning rate from $\eta_0 = 0.0$ to $\eta_{max} = 2\text{e-}5$ for 0.5% of the training iterations, then redecaying it with a cosine scheduler to reach $\eta_{min} = 0.1 \cdot \eta_{max}$ at the end of 2-epochs training, stopping it after 1 epoch due to clear signs of overfitting. Due to limited resources and budget TableLlama has been fine-tuned with LoRA [31] following [15] with a micro batch-size= 1, 64 gradients accumulation steps, LoRA-rank = 8, LoRA-α = 16, without any dropout or weight-decay. We have fine-tuned the Alligator's second model with the default hyperparameters as specified in [6].

Technical infrastructure. Both test and fine-tuning for TableLlama and TURL has run on a single NVIDIA A100-80GB; Alligator on an AMD EPYC-Milan Processor with 8 cores and 24GB of RAM, while Dagobah on an Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz with 32 cores and 96GB of RAM.

5 Results and discussion

We first focus on the main results, then we discuss evidence from ablation studies.

5.1 Main results

Table 4 reports the performances achieved by the four different approaches on the test data both for the pre-trained and fine-tuned models. We observe that Alligator excels on HT-R1, HT-R2 and WikidataTablesR1 SemTab datasets, it performs poorly on both BioDiv-red and TURL-2K-red-LamAPI while performing discretely on 2T-red dataset w.r.t. TableLlama. The opposite is observed for TableLlama and TURL, with TableLlama achieving generally higher accuracy than TURL. This trend can be explained by the fact that HT-R1, HT-R2 and WikidataTables-R1 are considered IN data for Alligator, while both BioDiv and TURL-2K-red-LamAPI, being OOD and MOOD data respectively, contain tables coming from different specific domains, with misspelled, repeated and abbreviated mentions, whose heterogeneity is difficult to capture with handcrafted features. Surprisingly, both TURL and TableLlama excel on BioDiv, even though it's considered OOD for both approaches: we hypothesise that the enormous and general pre-training knowledge retained by these models explains this excellence. The poor performance achieved by TURL on all the SemTab data, considered MOOD for it, can be explained by the fact that TURL is already fine-tuned on web tables, which are generally coming from a different data distribution than the ones of the SemTab challenge. Interestingly, we observed that both TableLlama and TURL underperforms themselves on the TURL-2K-LamAPI dataset: we argue that the drop in performances, especially for TURL, is due to the increased number of candidates we have retrieved with LamAPI (see the ablation study in section 5.2).

Figure 4: Accuracies achieved by TableLlama and TURL w.r.t. the number of candidates.

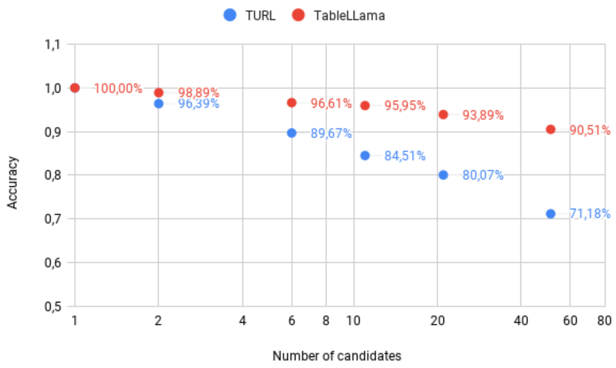
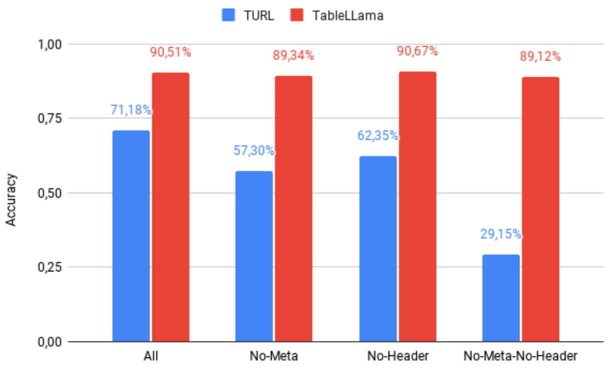


Figure 5: Accuracies achieved by TableLlama and TURL w.r.t. the presence of table's metadata.



MOOD fine-tuning. Table 4 reports also the performances of TURL, TableLlama and Alligator after the MOOD fine-tuning. Thanks to the "continual finetuning" inspired by [35] TableLlama slightly increases its performances on (previously) MOOD and OOD (BioDiv-red) data, with no clue of suffering from catastrophic forgetting on its IN data (TURL-2K-red-LamAPI). TURL gains the most from the MOOD fine-tuning (≈+36% on WikidataTablesR1) but suffers a severe drop in performance on both IN and OOD data, confirming the catastrophic forgetting suffered by models after fine-tuning [28]. Alligator achieves great performances on 2T-red, BioDiv-red and TURL-2K-red-LamAPI after MOOD fine-tuning, without suffering a huge drop in performance (apart for HT-R2, where it loses 10% points), demonstrating a good generalization pattern.

Computational performance. Figure 2 and Figure 3 report the overall elapsed time for the four tested approaches per dataset and the overall memory occupied by each approach per dataset (TURL and TableLlama's occupied memory refers to the A100-80GB GPU memory, while the Alligator and Dagobah's memory refers to the overall RAM of the machine hosting the algorithm). Both time and occupied memory are on a logarithmic scale. If TURL is the fastest and the lightest from both time and occupied memory perspectives, and TableLlama is the slowest and heaviest, Alligator strikes as a good compromise w.r.t. to execution time and occupied memory. Dagobah, on the other hand, is completely out-of-bounds, especially if one is bounded by resource or budget constraints.

5.2 Ablation studies

To explain the drop in performances observed for TURL and TableLlama on the TURL-2K-red-LamAPI dataset, we have measured the accuracy w.r.t. the number of candidates per mention, with the intuition that the higher the number of candidates the lower the performances. Figure 4 reports the accuracy of TURL and TableLlama given a different number of candidates per mention, considering also the correct one. Our intuitions are empirically confirmed by observing a drop in performance for both approaches, with a more severe one for TURL. This could happen because TURL aggregates the candidates of every mention in a table and passes them through a Transformer, increasing the context length to at most $O(NK)$, where N is the number of mentions in a table and K is the maximum number of candidates retrieved per mention.

We ran an additional ablation study to measure the impact of the table's metadata (*e.g.*, the title of the Wikipedia page the table is found in, the section title or the table caption to name a few) on the final accuracy achieved by both TURL and TableLlama, testing both on the TURL-2K-red-LamAPI dataset, the only dataset with table metadata, with i) *No-Meta*, *i.e.*, a setting where the page title, section title and the table caption are removed; ii) *No-Header*, *i.e.*, the table header is changed to $[\text{col}0, \text{col}1, ..., \text{col}N]$ and iii) *No-Meta-No-Header*, *i.e.*, the combination of both i) and ii). From Figure 5, we observe that TURL is heavily dependent on the table metadata, with a severe drop in performance when both metadata and header are removed. On the other hand, TableLlama is almost unaffected by removing metadata from the prompt, indicating a higher generalisation capability and a greater focus on the context provided by the table itself rather than by the metadata).

5.3 Discussion

Generative GTUM approaches with a high number of parameters seem to have interesting properties in terms of accuracy, generalisation, and robustness to the number of candidates that are processed and the table metadata, especially on MOOD and OOD data; however, specific STI approaches trained on in-domain data, and optionally fine-tuned on MOOD data, using a tiny fraction of these parameters can still outperform generalistic approaches with speed higher by an order of magnitude; these results may suggest that generative GTUM approaches are at the moment the best choice for processing small tables, while more specific entity disambiguation approaches may still be a better fit for applications on large business data. Generative approaches like TableLlama seem more promising than encoder-based ones in terms of performance, with a negligible risk of hallucinations; however, the comparison considered models of uncomparable size (7B vs 300M for TableLlama and TURL resp.), with TURL being the fastest approach among those that were tested. Regarding scalability, long context allows the encapsulation of large tables with a high number of candidates, but some tables cannot fit in the context, and when they do, the computation time and occupied memory increase proportionally to the table size: considering a small context table (few rows above and below the row containing the mention to be linked) to be fed to the model could be effective. On the budget side, training of models of the size of TableLlama has still enormous costs (48 A100-80GB for 9 training days), so devising generative methods based on smaller LLMs, *e.g.*, Phi [5], could be an interesting research direction, although more sophisticated approaches may be needed to achieve the same level of generalisation and reliability. Furthermore, TableLlama and TURL are not NIL aware and TableLlama, in particular, cannot return confidence scores associated with cell annotations, which may be useful when using these approaches to support revision: devising methods to estimate the uncertainty of labels computed by generative models may be an interesting research direction.

6 Conclusions and Future Works

STI, the process of annotating tabular data with information from background KGs, is proposed to support the understanding, interpretation and labeling of tables. Among the STI's tasks, CEA, *i.e.*, matching cell values to entities in the KG, is particularly relevant to support additional downstream transformation, integration, and enrichment processes, and is particularly subject to scalability constraints, e.g., when applied to tables with a large number of rows. LLMs pretrained with a vast amount of data have been applied to STI and CEA, complementing previous approaches based on heuristic and featured-based ML approaches. However, these different families of approaches have not been exhaustively examined on a common ground. In this work, we tackled this gap by selecting four representative approaches and comparatively evaluating them in terms of accuracy, generalisability, time, and memory requirements to better study their strengths and limitations, as well as their potential applications to different scenarios. We defined different evaluation settings, *i.e.*, “*in-domain*", “*out-of-domain*" and “*moderately out-of-domain*" for a better analysis of generalisability.

Our experiments suggest that an approach like TableLlama, based on a large generative LLM excels in accuracy and generalisation, as demonstrated by the results on MOOD and OOD data (see Table 4), at the price of an excessive execution and training time. TURL, an encoder-only model based on TinyBERT, is the most efficient, but lacks on generalisation capabilities: however, fine-tuning using data from similar distributions can lead to improvements with the new data at the price of a drop with pre-train data. Evaluating the impact of a larger encoder-based LLM on an approach like TURL could be an interesting research direction. However, our experiments suggest that specific STI models like Alligator, despite their limited number of parameters can still outperform generalistic models in IN-domain settings with a huge gain in efficiency.

Future works include the possibility to train a smaller and cheaper LLM-based model, *e.g.*, Phi [5], while enabling also the handling of NIL entities and a score associated with cell annotations. Another possible direction is to let both TURL and TableLlama be cell-based instead of table-based to reduce the occupied memory by both approaches and to enable standard augmentation techniques for the former.

The datasets, procedure source code for their creation, and evaluation code will be publicly accessible upon acceptance.

References

[1] Abdelmageed, N., Chen, J., Cutrona, V., Efthymiou, V., Hassanzadeh, O., Hulsebos, M., Jiménez-Ruiz, E., Sequeda, J., Srinivas, K.: Results of semtab 2022. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching **3320** (2022)

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

[3] Abdelmageed, N., Schindler, S.: Jentab meets semtab 2021's new challenges. In: SemTab@ ISWC. pp. 42–53 (2021)

[4] Abdelmageed, N., Schindler, S., König-Ries, B.: Biodivtab: A table annotation benchmark based on biodiversity research data. In: SemTab@ ISWC. pp. 13–18 (2021)

[5] Abdin, M., Jacobs, S.A., Awan, A.A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., Benhaim, A., Bilenko, M., Bjorck, J., Bubeck, S., Cai, M., Mendes, C.C.T., Chen, W., Chaudhary, V., Chopra, P., Giorno, A.D., de Rosa, G., Dixon, M., Eldan, R., Iter, D., Garg, A., Goswami, A., Gunasekar, S., Haider, E., Hao, J., Hewett, R.J., Huynh, J., Javaheripi, M., Jin, X., Kauffmann, P., Karampatziakis, N., Kim, D., Khademi, M., Kurilenko, L., Lee, J.R., Lee, Y.T., Li, Y., Liang, C., Liu, W., Lin, E., Lin, Z., Madan, P., Mitra, A., Modi, H., Nguyen, A., Norick, B., Patra, B., Perez-Becker, D., Portet, T., Pryzant, R., Qin, H., Radmilac, M., Rosset, C., Roy, S., Ruwase, O., Saarikivi, O., Saied, A., Salim, A., Santacroce, M., Shah, S., Shang, N., Sharma, H., Song, X., Tanaka, M., Wang, X., Ward, R., Wang, G., Witte, P., Wyatt, M., Xu, C., Xu, J., Yadav, S., Yang, F., Yang, Z., Yu, D., Zhang, C., Zhang, C., Zhang, J., Zhang, L.L., Zhang, Y., Zhang, Y., Zhang, Y., Zhou, X.: Phi-3 technical report: A highly capable language model locally on your phone (2024)

[6] Avogadro, R., Ciavotta, M., De Paoli, F., Palmonari, M., Roman, D.: Estimating link confidence for human-in-the-loop table annotation. In: 2023 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT). pp. 142–149 (2023)

[7] Avogadro, R., Cremaschi, M.: Mantistable v: A novel and efficient approach to semantic table interpretation. In: SemTab@ ISWC. pp. 79–91 (2021)

[8] Avogadro, R., Cremaschi, M., D'adda, F., De Paoli, F., Palmonari, M.: Lamapi: a comprehensive tool for string-based entity retrieval with type-base filters. In: 17th ISWC workshop on ontology matching (OM). p. Online (2022)

[9] Azzi, R., Diallo, G., Jiménez-Ruiz, E., Hassanzadeh, O., Efthymiou, V., Chen, J., Srinivas, K.: Amalgam: making tabular dataset explicit with knowledge graph. In: SemTab@ ISWC. pp. 9–16 (2020)

[10] Baazouzi, W., Kachroudi, M., Faiz, S.: Kepler-asi at semtab 2021. In: SemTab@ ISWC. pp. 54–67 (2021)

[11] Bhagavatula, C.S., Noraset, T., Downey, D.: Tabel: Entity linking in web tables. In: The Semantic Web - ISWC 2015. pp. 425–441 (2015)

[12] Chen, J., Jiménez-Ruiz, E., Horrocks, I., Sutton, C.: Colnet: Embedding the semantics of web tables for column type prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 29–36 (2019)

[13] Chen, S., Karaoglu, A., Negreanu, C., Ma, T., Yao, J.G., Williams, J., Gordon, A., Lin, C.Y.: Linkingpark: An integrated approach for semantic table interpretation. In: SemTab@ ISWC (2020)

[14] Chen, S., Karaoglu, A., Negreanu, C., Ma, T., Yao, J.G., Williams, J., Jiang, F., Gordon, A., Lin, C.Y.: Linkingpark: An automatic semantic table interpretation system. Journal of Web Semantics **74**, 100733 (2022)

[15] Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., Jia, J.: Longlora: Efficient fine-tuning of long-context large language models (2024)

[16] Clark, K., Luong, M., Le, Q.V., Manning, C.D.: ELECTRA: pre-training text encoders as discriminators rather than generators. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=r1xMH1BtvB

[17] Cremaschi, M., Avogadro, R., Barazzetti, A., Chieregato, D., Jiménez-Ruiz, E.: Mantistable se: an efficient approach for the semantic table interpretation. In: SemTab@ ISWC. pp. 75–85 (2020)

[18] Cremaschi, M., Avogadro, R., Chieregato, D.: Mantistable: an automatic approach for the semantic table interpretation. SemTab@ ISWC **2019**, 15–24 (2019)

[19] Cremaschi, M., Avogadro, R., Chieregato, D.: s-elbat: a semantic interpretation approach for messy table-s. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab), CEUR-WS. org (2022)

[20] Cremaschi, M., De Paoli, F., Rula, A., Spahiu, B.: A fully automated approach to a complete semantic table interpretation. Future Generation Computer Systems **112**, 478 – 500 (2020)

[21] Cutrona, V., Bianchi, F., Jiménez-Ruiz, E., Palmonari, M.: Tough tables: Carefully evaluating entity linking for tabular data. In: Pan, J.Z., Tamma, V., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web – ISWC 2020. pp. 328–343. Springer International Publishing, Cham (2020)

[22] Dasoulas, I., Yang, D., Duan, X., Dimou, A.: Torchictab: Semantic table annotation with wikidata and language models. In: CEUR Workshop Proceedings. pp. 21–37.

CEUR Workshop Proceedings (2023)

[23] Deng, X., Sun, H., Lees, A., Wu, Y., Yu, C.: Turl: Table understanding through representation learning. ACM SIGMOD Record **51**(1), 33–40 (2022)

[24] Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019)

[25] Efthymiou, V., Hassanzadeh, O., Rodriguez-Muro, M., Christophides, V.: Matching web tables with knowledge base entities: From entity lookups to entity embeddings. In: The Semantic Web – ISWC 2017. pp. 260–277 (2017)

[26] Efthymiou, V., Hassanzadeh, O., Rodriguez-Muro, M., Christophides, V.: Matching web tables with knowledge base entities: From entity lookups to entity embeddings. In: The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21–25, 2017, Proceedings, Part I. p. 260–277. Springer-Verlag, Berlin, Heidelberg (2017)

[27] Fang, X., Xu, W., Tan, F.A., Hu, Z., Zhang, J., Qi, Y., Sengamedu, S.H., Faloutsos, C.: Large language models (LLMs) on tabular data: Prediction, generation, and understanding - a survey. Transactions on Machine Learning Research (2024), https://openreview.net/forum?id=IZnrCGF9WI

[28] Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks (2015), https://arxiv.org/abs/1312.6211

[29] Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch sgd: Training imagenet in 1 hour (2018)

[30] Hassanzadeh, O., Abdelmageed, N., Efthymiou, V., Chen, J., Cutrona, V., Hulsebos, M., Jiménez-Ruiz, E., Khatiwada, A., Korini, K., Kruit, B., et al.: Results of semtab 2023. In: CEUR Workshop Proceedings. vol. 3557, pp. 1–14 (2023)

[31] Hu, E.J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=nZeVKeeFYf9

[32] Huynh, V.P., Chabot, Y., Labbé, T., Liu, J., Troncy, R.: From heuristics to language models: A journey through the universe of semantic table interpretation with dagobah. SemTab (2022)

[33] Huynh, V.P., Liu, J., Chabot, Y., Deuzé, F., Labbé, T., Monnin, P., Troncy, R.: Dagobah: Table and graph contexts for efficient semantic annotation of tabular data. In: SemTab@ ISWC. pp. 19–31 (2021)

[34] Huynh, V.P., Liu, J., Chabot, Y., Labbé, T., Monnin, P., Troncy, R.: Dagobah: Enhanced scoring algorithms for scalable annotations of tabular data. In: SemTab@ ISWC. pp. 27–39 (2020)

[35] Ibrahim, A., Thérien, B., Gupta, K., Richter, M.L., Anthony, Q., Lesort, T., Belilovsky, E., Rish, I.: Simple and scalable strategies to continually pre-train large language models (2024)

[36] Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., Liu, Q.: TinyBERT: Distilling BERT for natural language understanding. In: Cohn, T., He, Y., Liu, Y. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 4163–4174. Association for Computational Linguistics, Online (Nov 2020)

[37] Jiménez-Ruiz, E., Hassanzadeh, O., Efthymiou, V., Chen, J., Srinivas, K.: Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems. In: The Semantic Web. pp. 514–530. Springer, Cham (2020)

[38] Jiménez-Ruiz, E., Hassanzadeh, O., Efthymiou, V., Chen, J., Srinivas, K.: Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems. In: Harth, A., Kirrane, S., Ngonga Ngomo, A.C., Paulheim, H., Rula, A., Gentile, A.L., Haase, P., Cochez, M. (eds.) The Semantic Web. pp. 514–530. Springer International Publishing, Cham (2020)

[39] Kandpal, N., Deng, H., Roberts, A., Wallace, E., Raffel, C.: Large language models struggle to learn long-tail knowledge. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (eds.) Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 15696–15707. PMLR (23–29 Jul 2023)

[40] Kirk, R., Mediratta, I., Nalmpantis, C., Luketina, J., Hambro, E., Grefenstette, E., Raileanu, R.: Understanding the effects of RLHF on LLM generalisation and diversity. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=PXD3FAVHJT

[41] Kong, K., Zhang, J., Shen, Z., Srinivasan, B., Lei, C., Faloutsos, C., Rangwala, H., Karypis, G.: Opentab: Advancing large language models as open-domain table reasoners. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=Qa0ULgosc9

[42] Korini, K., Bizer, C.: Column type annotation using chatgpt. arXiv preprint arXiv:2306.00745 (2023)

[43] Korini, K., Peeters, R., Bizer, C.: Sotab: The wdc schema. org table annotation benchmark. Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab), CEUR-WS. org (2022)

[44] Kruit, B., Boncz, P., Urbani, J.: Extracting novel facts from tables for knowledge graph completion. In: The Semantic Web – ISWC 2019. pp. 364–381f. Springer International Publishing, Cham (2019)

[45] Li, P., He, Y., Yashar, D., Cui, W., Ge, S., Zhang, H., Fainman, D.R., Zhang, D., Chaudhuri, S.: Table-gpt: Table-tuned gpt for diverse table tasks (2023)

[46] Li, T., Shetty, S., Kamath, A., Jaiswal, A., Jiang, X., Ding, Y., Kim, Y.: Cancergpt for few shot drug pair synergy prediction using large pretrained language models. NPJ Digital Medicine **7**(1), 40 (2024)

[47] Li, Y., Li, J., Suhara, Y., Doan, A., Tan, W.C.: Deep entity matching with pre-trained language models. VLDB (2020)

[48] Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. Proc. VLDB Endow. **3**(1-2), 1338–1347 (Sep 2010)

[49] Liu, J., Chabot, Y., Troncy, R., Huynh, V.P., Labbé, T., Monnin, P.: From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods. Journal of Web Semantics **76**, 100761 (2023). https://doi.org/https://doi.org/10.1016/j.websem.2022.100761, https://www.sciencedirect.com/science/article/pii/S1570826822000452

[50] Liu, J., Huynh, V.P., Chabot, Y., Troncy, R.: Radar station: Using kg embeddings for semantic table interpretation and entity disambiguation. In: ISWC 2022, October 23–27, 2022. pp. 498–515. Springer (2022)

[51] Morikawa, H.: Semantic table interpretation using lod4all. SemTab@ ISWC **2019**, 49–56 (2019)

[52] Mulwad, V., Finin, T., Joshi, A.: Semantic message passing for generating linked data from tables. In: The Semantic Web – ISWC 2013. pp. 363–378. Springer Berlin Heidelberg (2013)

[53] Mulwad, V., Finin, T., Syed, Z., Joshi, A.: T2ld: Interpreting and representing tables as linked data. In: ISWC Posters & Demonstrations Track. pp. 25–28. ISWC-PD'10, CEUR-WS.org, Aachen, Germany, Germany (2010)

[54] Mulwad, V., Finin, T.W., Joshi, A.: Automatically generating government linked data from tables. In: AAAI 2011 (2011)

[55] Nahid, M.M.H., Rafiei, D.: Tabsqlify: Enhancing reasoning capabilities of llms through table decomposition (2024), https://arxiv.org/abs/2404.10150

[56] Nguyen, P., Kertkeidkachorn, N., Ichise, R., Takeda, H.: Mtab: matching tabular data to knowledge graph using probability models. arXiv preprint arXiv:1910.00246 (2019)

[57] Nguyen, P., Yamada, I., Kertkeidkachorn, N., Ichise, R., Takeda, H.: Mtab4wikidata at semtab 2020: Tabular data annotation with wikidata. SemTab@ ISWC **2775**, 86–95 (2020)

[58] Nguyen, P., Yamada, I., Kertkeidkachorn, N., Ichise, R., Takeda, H.: Semtab 2021: Tabular data annotation with mtab tool. In: SemTab@ ISWC. pp. 92–101 (2021)

[59] Oliveira, D., d'Aquin, M.: Adog-annotating data with ontologies and graphs. SemTab@ ISWC **2019**, 1–6 (2019)

[60] Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: A roadmap. IEEE Transactions on Knowledge and Data Engineering p. 1–20 (2024)

[61] Peeters, R., Bizer, C.: Using chatgpt for entity matching. arXiv preprint arXiv:2305.03423 (2023)

[62] Pham, M., Alse, S., Knoblock, C.A., Szekely, P.: Semantic labeling: a domain-independent approach. In: The Semantic Web–ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part I 15. pp. 446–462. Springer (2016)

[63] Rao, D., McNamee, P., Dredze, M.: Entity Linking: Finding Extracted Entities in a Knowledge Base, pp. 93–115. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

[64] Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009). pp. 147–155. Association for Computational Linguistics, Boulder, Colorado (Jun 2009)

[65] Ritze, D., Lehmberg, O., Bizer, C.: Matching html tables to dbpedia. In: 5th International Conference on Web Intelligence, Mining and Semantics. pp. 10:1–10:6. WIMS '15, ACM, New York, NY, USA (2015)

[66] Shigapov, R., Zumstein, P., Kamlah, J., Oberländer, L., Mechnich, J., Schumm, I.: bbw: Matching csv to wikidata via meta-lookup. In: CEUR Workshop Proceedings. vol. 2775, pp. 17–26. RWTH (2020)

[67] Steenwinckel, B., Vandewiele, G., De Turck, F., Ongenae, F.: Csv2kg: Transforming tabular data into semantic knowledge. SemTab, ISWC Challenge (2019)

[68] Suhara, Y., Li, J., Li, Y., Zhang, D., Demiralp, c., Chen, C., Tan, W.C.: Annotating columns with pre-trained language models. In: Proceedings of the 2022 International Conference on Management of Data. p. 1493–1503. SIGMOD '22, Association for Computing Machinery, New York, NY, USA (2022)

[69] Sui, Y., Zhou, M., Zhou, M., Han, S., Zhang, D.: Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In: Proceedings of the 17th ACM International Conference on Web Search and Data Mining. p. 645–654. WSDM '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3616855.3635752, https://doi.org/10.1145/3616855.3635752

[70] Syed, Z., Finin, T., Mulwad, V., Joshi, A.: Exploiting a web of semantic data for interpreting tables. In: Proceedings of the Second Web Science Conference. vol. 5 (2010)

[71] Thawani, A., Hu, M., Hu, E., Zafar, H., Divvala, N.T., Singh, A., Qasemi, E., Szekely, P.A., Pujara, J.: Entity linking to knowledge graphs to infer column types and properties. SemTab@ ISWC **2019**, 25–32 (2019)

[72] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C.C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P.S., Lachaux, M.A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X.E., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T.: Llama 2: Open foundation and fine-tuned chat models (2023)

[73] Tyagi, S., Jimenez-Ruiz, E.: Lexma: Tabular data to knowledge graph matching using lexical techniques. In: CEUR Workshop Proceedings. vol. 2775, pp. 59–64 (2020)

[74] Wang, J., Liang, Y., Meng, F., Sun, Z., Shi, H., Li, Z., Xu, J., Qu, J., Zhou, J.: Is ChatGPT a good NLG evaluator? a preliminary study. In: Dong, Y., Xiao, W., Wang, L., Liu, F., Carenini, G. (eds.) Proceedings of the 4th New Frontiers in Summarization Workshop. pp. 1–11. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.newsum-1.1, https://aclanthology.org/2023.newsum-1.1

[75] Wang, Z., Zhang, H., Li, C.L., Eisenschlos, J.M., Perot, V., Wang, Z., Miculicich, L., Fujii, Y., Shang, J., Lee, C.Y., Pfister, T.: Chain-of-table: Evolving tables in the reasoning chain for table understanding. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=4L0xnS4GQM

[76] Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A.W., Lester, B., Du, N., Dai, A.M., Le, Q.V.: Finetuned language models are zero-shot learners. In: International Conference on Learning Representations (2022)

[77] Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., Mann, G.: Bloomberggpt: A large language model for finance (2023)

[78] Yang, L., Shen, S., Ding, J., Jin, J.: Gbmtab: A graph-based method for interpreting noisy semantic table to knowledge graph. In: SemTab@ ISWC. pp. 32–41 (2021)

[79] Zhang, S., Meij, E., Balog, K., Reinanda, R.: Novel entity discovery from web tables. In: Proceedings of The Web Conference 2020. p. 1298–1308. WWW '20, Association for Computing Machinery, New York, NY, USA (2020)

[80] Zhang, T., Yue, X., Li, Y., Sun, H.: Tablellama: Towards open large generalist models for tables (2023)

[81] Zhang, Y., Zhang, M., Yuan, H., Liu, S., Shi, Y., Gui, T., Zhang, Q., Huang, X.: Llmeval: A preliminary study on how to evaluate large language models. Proceedings of the AAAI Conference on Artificial Intelligence **38**(17), 19615–19622 (Mar 2024). https://doi.org/10.1609/aaai.v38i17.29934, https://ojs.aaai.org/index.php/AAAI/article/view/29934

[82] Zhang, Z.: Effective and efficient semantic table interpretation using tableminer+. Semantic Web **8**(6), 921–957 (2017)

[83] Zheng, M., Yang, H., Jiang, W., Lin, Z., Lyu, Y., She, Q., Wang, W.: Chain-of-thought reasoning in tabular language models. In: Bouamor, H., Pino, J., Bali, K. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 11006–11019. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.findings-emnlp.734, https://aclanthology.org/2023.findings-emnlp.734

[84] Zhong, L., Wang, Z.: Can llm replace stack overflow? a study on robustness and reliability of large language model code generation. Proceedings of the AAAI Conference on Artificial Intelligence **38**(19), 21841–21849 (Mar 2024). https://doi.org/10.1609/aaai.v38i19.30185, https://ojs.aaai.org/index.php/AAAI/article/view/30185