



How good are LLMs in disambiguating entities in tabular data? A comprehensive study[☆]

Federico Belotti ^a *, Marco Cremaschi ^a, Fabio Dadda ^a, Roberto Avogadro ^b, Matteo Palmonari ^a

^a University of Milano-Bicocca, viale Sarca 336, Milan, 20126, Italy

^b SINTEF, Forskningsveien 1, Oslo, 0373, Norway

ARTICLE INFO

Dataset link: <https://huggingface.co/datasets/belotrico/evaluating-llms-on-tabular-data>

Keywords:

Semantic table interpretation
Large language models
Entity disambiguation
Entity linking
Knowledge graph

ABSTRACT

Tables are crucial containers of information, but understanding their meaning may be challenging. 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 ones. In the last period, the advent of Large Language Models (LLMs) has led to a new category of approaches for table annotation. However, these approaches have not been consistently evaluated on a common ground, making evaluation and comparison difficult. This work uniquely compares Semantic Table Interpretation (STI) approaches with generative and encoder-only LLMs on diverse datasets. In particular, we conduct an extensive evaluation of four STI state-of-the-art (SOTA) approaches — Alligator (formerly s-elBat), TURL, TableLlama, and DAGOBAB (the latter with a partial evaluation due to its high computational demands); Alligator and DAGOBAB belong to the family of heuristic-based algorithms, while TURL and TableLlama are respectively encoder-only and decoder-only LLMs. We also include in the evaluation both GPT-4o and GPT-4o-mini, since they excel in various public benchmarks. The primary objective is to measure the ability of these approaches to solve the entity disambiguation task concerning both the performance achieved on a common-ground evaluation setting and the computational and cost requirements involved, either monetary or in terms of computational resources, with the ultimate aim of charting new research paths in the field.

1. Introduction

Tables are commonly used to create, organise, and share information in various knowledge-intensive processes and applications in both business and science. Linking values occurring in the table with entities from a background Knowledge Graph (KG) is an entity reconciliation task with several impactful applications to semantic data management and semantic system engineering [1].

First, linking entities in tables helps automate processes developed to ingest tabular data into semantic data management frameworks, such as KG construction and extension (exemplified in the top two processes in Fig. 1). In this context, entity linking is usually considered part of broader table-to-graph matching solutions that interpret the elements of a table using the entity identifiers in the background KG and its vocabulary. Background KGs are typically large, represented in RDF, and contain encyclopedic knowledge, e.g., Wikidata [2] and DBpedia [3]; domain-specific KGs can also be considered [4]. Different research communities have

[☆] This article is part of a Special issue entitled: 'LLMs and KGs for Semantics-driven Sys Engineering - Woo' published in Data & Knowledge Engineering.

* Corresponding author.

E-mail addresses: federico.belotti@unimib.it (F. Belotti), marco.cremaschi@unimib.it (M. Cremaschi), fabio.dadda@unimib.it (F. Dadda), roberto.avogadro@sintef.no (R. Avogadro), matteo.palmonari@unimib.it (M. Palmonari).

<https://doi.org/10.1016/j.datak.2026.102596>

Received 29 March 2025; Received in revised form 15 March 2026; Accepted 18 March 2026

Available online 20 March 2026

0169-023X/© 2026 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

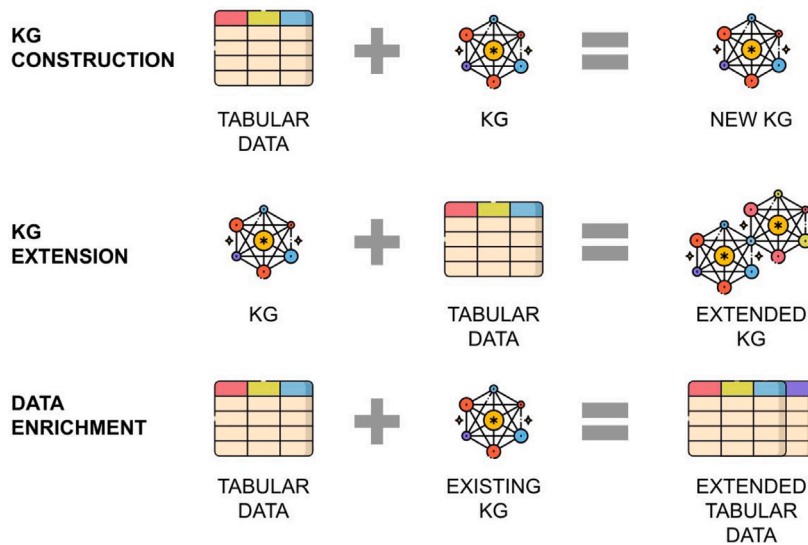


Fig. 1. Semantics-driven system engineering tasks supported by tabular data annotation approaches.

considered table-to-graph matching tasks under slightly different formulations, such as Semantic Table Interpretation (STI) [5] - from which we borrow our terminology, semantic labelling [6], and table annotation [7]. Entity linking corresponds to the Cell-Entity Annotation (CEA) task in STI: cell values that occur in columns referring to entities (as opposed, e.g., to columns representing numbers) are annotated with entity identifiers. Most of the approaches for the CEA task decompose linking into two main sub-tasks [8]: in Entity Retrieval (ER), a set of candidate entities is gathered for each mention, often accompanied by an initial score and ranking; Entity Disambiguation (ED) then selects the most suitable candidate and, in some cases, determines whether a mention should be linked at all [9]. Two other key table-to-graph matching tasks are Column-Type Annotation (CTA) - columns are annotated with class labels, and Column-Property Annotation (CPA) - pairs of columns are annotated with properties [5,10]. Annotations carry sufficient information to automate the extraction of graph data from the input table, using the entity identifiers of the background KG, its vocabulary, or both Cremaschi et al. [11].

Second, entity linking (CEA, as defined above) supports semantics-driven system engineering processes developed to enrich tabular data with rich semantic features, and eventually augment their content [12,13]. We refer to these processes as semantic data enrichment (exemplified in the bottom process in Fig. 1). Several examples of these semantic data enrichment processes can be found in the literature. For example, Cutrona et al. [14] link spatial references to Geonames identifiers in the context of a weather-based marketing campaign optimisation system; Avogadro et al. [15] link public organisation names to Wikidata entities to enrich a public procurement KG; Novalija et al. [16] link affiliations in GitHub data to Wikidata identifiers to enrich data and eventually build a KG to monitor artificial intelligence development. In these examples, links are used to improve the quality of data (e.g., a link to a Wikidata entity is considered evidence for the existence of a proper public organisation [15]), integrate data (e.g., different GitHub repositories are indirectly connected when affiliations of owners map to the same organisation in Wikidata [16]), fetch data from the background KG (e.g., fetching geocoordinates from Geonames [14]).

The application of entity linking to semantic-based system engineering underpins the effort to build tools that support interactive annotation of tabular data using automated linking algorithms. These tools are developed to help data engineers transform tabular data into graph data and/or enrich tabular data using KGs. Examples of such tools, each one with different capabilities, include OpenRefine,¹ *Kgextension*, MAGIC [17], DAGOBAN-UI [18], and SemT-UI [13,19].

To sum up, entity linking on tabular data, i.e., the CEA task as defined in STI, serves as a building block in several fully automated or interactive downstream data management processes based on semantics and KGs. These processes span from harmonising identifiers and vocabulary when transforming tabular data into graph data to exploiting semantic links to improve the quality of the data, augment it, and support integration across tables. However, when linking entities in tables in these semantic-driven system engineering processes, the accuracy of links matters. Even when using interactive tools, data engineers cannot revise all links returned by automated methods, especially on large tables [9,13]. The CEA task is challenging due to the diverse structures of tables, which can vary greatly depending on their context of origin. 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 of thousands 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) [5]. This diversity underscores the need for CEA algorithms to

¹ <https://openrefine.org/>.

Table 1

Comparison of TURL, TableLlama, and Alligator for the CEA task. “MLM” denotes Masked Language Modelling, and “MER” denotes Masked Entity Retrieval [20].

	TURL [20]	TableLlama [22]	Alligator [9]
Model	TinyBERT [27] equipped with table structure-awareness (visibility matrix)	Llama2 with LongLora [28] for extended context	Feature-based ML approach with two Neural Network (NN)s for scoring
Input	Table comprised of only the mentions to be linked, optional metadata and candidates for every mentions	Serialized table with instruction, optional metadata, mention to be linked and the list of candidates, limited to 8192 tokens	Whole table (candidates are retrieved on-the-fly thanks to <i>LamAPI</i>)
Objective	Pre-training with MLM and MER; task-specific fine-tuning on CEA	Multi-task instruction tuning for CEA, CTA, CPA, and other table-related tasks	Feature-based binary cross-entropy with negatives sampling: $p(y_m = 1 F_m) \geq$ $p(y_m = 0 F_w; w \in \mathcal{N}(m))^a$
Output	Probability distribution over candidates	(Auto-regressively) Extracts candidate directly from list	Score in [0, 1] for every candidate
Limitations	Input table limited to mentions to be linked, no NIL-awareness	Limited to 8192 tokens, no score over candidates, no NIL-awareness	No semantic-related features, biased towards KG matching

^a Given a mention m and its correct associated entity in the knowledge graph, $p(y_m = 1 | F_m)$ represents the probability that the features F_m correctly identify the correct candidate. For each incorrect candidate w in the set $\mathcal{N}(m)$, $p(y_m = 0 | F_w)$ represents the probability that the features F_w correctly indicate that w is not the correct candidate.

be evaluated across multiple dimensions, ensuring their effectiveness in different scenarios. Moreover, efficiency and generalisation are essential qualities for such algorithms.

A first generation of methods for CEA has exploited different matching heuristics, traditional machine learning approaches based on engineered features (hereafter “feature-based ML”), or a combination of both Liu et al. [5]. 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 and other tasks, e.g., question answering, schema augmentation, row population, and fact verification among others. The first remarkable example is TURL [20], which is based on an adaptation of BERT [21] to consider the tabular structure, additionally fine-tuned to execute specific tasks (including ED). The latest of these models, TableLlama [22], is based on the autoregressive Large Language Model (LLM) Llama 2-7B [23], which is fine-tuned with instruction tuning [24] to perform specific tasks. These approaches have been trained and tested on datasets that fit their generalistic ambition and reported as SOTA approaches on the considered test data.

1.1. Limitations in current studies

Within STI, both feature-based methods and GTUM approaches support entity disambiguation (ED), yet a comparative evaluation of these two families on a common experimental ground has not yet been conducted. Feature-based methods have primarily been evaluated within the International Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab) [25]. In this challenge, most tables are synthetically generated by querying Knowledge Graphs (KGs). These tables can be large and closely resemble those used in business processes but lack the surrounding textual descriptions typically found in web tables, such as those on Wikipedia. In contrast, GTUM approaches have been evaluated mainly on web-based tables [26]. These tables are relatively small, surrounded by rich contextual information, and crafted by humans for human use. While feature-based methods are designed to select from a wide range of candidate entities retrieved using coverage-maximising techniques, GTUM approaches have been tested with limited candidate lists obtained via the off-the-shelf Wikidata lookup service.² As a result, it is unclear which family of methods demonstrates better disambiguation capabilities under different conditions. Additionally, several critical aspects of CEA, e.g., efficiency and generalisation, have not been adequately addressed in previous studies. Furthermore, the latest generation of Large Language Models (LLMs), such as GPT-4o, which are reportedly trained for data analysis tasks³ and could potentially serve as off-the-shelf ED services, have not been considered in past comparisons. We believe these questions are crucial for understanding the current state of ED table algorithms and support knowledge engineers who must choose the most promising solution for a problem with specific features.

1.2. Contributions of this paper and outline

The study presented in this paper provides the first systematic and multi-dimensional analysis of the advantages and disadvantages of GTUM models and LLMs on the ED task, especially when compared to approaches specifically targeting CEA developed in the context of STI. Our analysis mainly considers the SOTA generalistic generative model TableLlama [22], its predecessor TURL [20],

² https://www.wikidata.org/wiki/Wikidata:REST_API.

³ <https://help.openai.com/en/articles/8437071-data-analysis-with-chatgpt>.

a BERT-based encoder-only transformer, Alligator [9,29] (formerly s-elBat [30]), a recent feature-based Machine Learning (ML) approach, and DAGOBAN, a top-performing heuristic-based system from the SemTab challenge (with a partial evaluation due to its high computational demands). These models represent four different inference mechanisms for ED, briefly detailed in Table 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 small neural networks to return confidence scores; DAGOBAN combines multiple heuristics and an ELECTRA-based cross-encoder, representing another well-established paradigm. To support this analysis we introduce an evaluation framework to test each approach in *in-domain*, *out-of-domain*, and *moderately out-of-domain* settings, as more precisely defined in Section 4.3. Using this framework, we evaluate the impact of fine-tuning strategies to test the generalisation capabilities of TURL, TableLlama, and Alligator. Finally, in specific settings, we also consider results obtained by GPT-4o and GPT-4o-mini.⁴ The last two models, trained on unknown data and based on an unknown number of parameters, can provide off-the-shelf ED services accessible via APIs. Consistent with our attention to data engineering aspects, we also estimate and compare monetary costs (in USD) and computational resource consumption (execution time and memory usage), draw conclusions in the form of recommendations and discuss the impact of our results when CEA methods are applied to semantic-driven data management tasks (see Section 6).

While we do not introduce novel entity linking methods, the novel findings presented in this paper add up to the important insights that scientists are accumulating on the capabilities of LLMs in solving specific data engineering and NLP tasks [31–38].

The key contributions of this paper can be summarised as follows:

1. We provide a direct comparison of state-of-the-art approaches that were previously evaluated on different datasets, now assessed under a unified evaluation framework.
2. We introduce a framework for evaluating generalisation capabilities based on in-domain, moderately-out-of-domain, and out-of-domain tests.
3. We conduct a comprehensive assessment that considers not only performance but also efficiency and cost—critical factors for real-world applications.
4. We evaluate two top-tier LLMs with different cost structures, GPT-4o and GPT-4o-mini, which have not been previously tested for this task.

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. Section 6 discusses the generalised implications of our findings for semantics management and semantics-driven system engineering. Finally, we conclude this paper and discuss the future direction in Section 7.

2. Related work

We remind that CEA is usually divided into two sub-tasks: *retrieval of candidate entities*, i.e., ER, and *entity disambiguation*, i.e., ED [8], where one or no candidate is selected as a link, usually after scoring and ranking the candidates. Since approaches based on language models use off-the-shelves retrieval services and focus on the ED task, in the discussion of related work we also focus on this task. For a more comprehensive discussion on STI we refer to existing surveys [5,11]. 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 language models or LLM.

Heuristic, ML and Probabilistic-based Approaches. The ED task in STI can be performed by applying multiple techniques while focusing on different inherent information [5,11]: (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 [39–55]. *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 [44,47,48,51,53–69].

Other methods employed are **ML techniques**. These techniques typically involve training a ML model on a labelled dataset where cells are annotated with their corresponding entities. The models, like Support Vector Machine (SVM) [70], Neural Network (NN) [44] and Random Forest [71], 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 [9,29]; 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) [42,72–75].

⁴ For reference: <https://openai.com/index/hello-gpt-4o/> and <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>.

In the current SOTA, several attempts to apply **LLMs-based approaches** 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* [76].

Starting from *encoder-based approaches*, Ditto [77] utilises Transformer-based language models to perform a slightly different task, i.e., entity-matching between different tables. Doduo [7] performs Column-Type Annotation (CTA) and CPA using a pre-trained language model, i.e., a fine-tuned BERT model on serialised tabular data. Column types (properties) are predicted using a dense layer followed by an output layer with a size corresponding to the number of column types (properties). DAGOBAB SL 2022 [68] employs an ELECTRA-based [78] 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 [79] is composed of two sub-systems: TorchicTab-Heuristic and TorchicTab-Classification, with the latter that utilises Doduo [7] internally.

Regarding the *decoder-only approaches*, Korini and Bizer [80] explored the CTA task by employing ChatGPT, performing experiments with diverse prompts tailored for the task using a subset of the SOTAB benchmark [81]. 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 [82]. 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 [83] for the financial domain or CancerGPT for the medical one [84]. Other works, instead, focused on table-specific tasks, especially reasoning [85–87], enhanced by Chain-of-Thought (CoT) [88,89]. Another decoder-based approach, TableGPT [90], 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 [20], which leverages a pre-trained TinyBERT [27] 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 [22], 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 [20] dataset, by prompting the LLM to retrieve the correct candidate given the serialised 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 [91].

Alongside advances in STI methods, the availability of comprehensive benchmarking resources has become increasingly important for systematic evaluation. *MammoTab 25* [92] is a recent large-scale dataset comprising approximately 839 000 Wikipedia tables extracted from English Wikipedia and annotated through Wikidata, with fine-grained metadata and tags that enable researchers to isolate specific semantic challenges (e.g., homonymy, NIL mentions, type selection). A curated subset serves as the primary test collection for SemTab 2025, and a public leaderboard benchmarking 19 open-weight LLMs provides a continuously updated yardstick for the community. The *MammoTab 25* evaluation is complementary to ours: it focuses exclusively on open-weight models (ranging from 0.5B to 123B parameters) and does not include proprietary LLMs such as GPT-4o or GPT-4o-mini; we discuss how its findings integrate with the results presented in this paper in Section 6.

3. Selected approaches

We select four STI approaches representative of different categories of algorithms that solve semantic tasks on tables, and especially CEA, each encoding a particular inductive bias. TableLlama is the first autoregressive LLM that is specifically instruction-tuned on tabular data and reports SOTA results [22]. Thanks to its extensive text pre-training and instruction tuning, TableLlama is biased towards text-based reasoning and instruction-following behaviour, with only indirect access to KG structure through textual descriptions. TURL, cited as previous SOTA in [22], is a BERT-based encoder-only model performing the three STI tasks, including CEA [20]. It is biased towards table structure and its pre-training on WikiTables, strongly relying on headers, captions and additional table metadata. 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 [9]. By design, it is biased towards KG structure and lexical/syntactic similarity (popularity, token overlap, row-level literal matching, column-type/property consistency), and favours entities that fit the KG neighbourhood of the row. The fourth approach is DAGOBAB, a heuristic-based algorithm that has been the winner of various rounds of the SemTab challenge in 2020 [48], 2021 [53] and 2022 [68] and is publicly available; due to its high computational demands, its evaluation is partial. As summarised in Table 1, the approaches also differ in terms of input representation, pre-training objectives, and explicit access to KG structure. Their precise details are discussed next.

While we focus our study only on the CEA task, 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.

TURL [20] 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 [27] 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.⁵ 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 [22], 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 [22] employs Llama2 [23] and instruction tuning [24] 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 [28] 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 it 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 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 applies standard pre-processing transformations (e.g., lowercase, space removing), with columns 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., its KG popularity, number of tokens and the syntactical similarity between the candidate label and the mention) or its comparison with the row cells content (e.g., trying to match related literals with the row content). *Local scoring* computes a matching score for each candidate, using a simple deep NN (denoted PN in [9]): the NN takes the local feature vectors as input and is trained to convert the features into 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 mention 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 (denoted RN in [9]) similar to the previous one predicts the updated normalised scores for all the candidates.⁶

DAGOBAB is an end-to-end STI system that annotates relational tables with entities, types, and relations from a target KG such as Wikidata. Its pipeline starts with table pre-processing, followed by an entity lookup step based on Elasticsearch indexes over KG labels and enriched aliases. For each candidate entity of a target cell, DAGOBAB computes a pre-score that combines (i) literal similarity between the cell value and the candidate's labels and (ii) a *graph-context score*. To compute this graph-context score, the system builds a small subgraph around each candidate by exploring one- and two-hop predicate paths in the KG and matches nodes in this subgraph against neighbouring cells in the same row. Each neighbouring cell is weighted by several factors: whether it contains an entity or a literal, its distance (in columns) from the first object column, and the *connectivity* of its column to the target column, i.e., how often a KG relation is observed between entities in those two columns across the table. Finally, Column-Type (CTA) and Column-Property (CPA) predictions are computed via majority voting over candidate entities. In our experiments, we only report a *partial evaluation* of DAGOBAB. The reason is practical rather than methodological: building a KG subgraph and computing soft context scores for every candidate entity is computationally expensive on a large KG like Wikidata, especially when many candidates are available per cell. On several of our datasets, this per-candidate graph-context construction led to very high runtimes and, in some cases, out-of-memory or timeout issues. We therefore restricted DAGOBAB to a subset of tables where it could be executed reliably and clearly marked its results as partial in our evaluation.

Table 2 lists the datasets used to pre-train the selected approaches; moreover, for TableLlama we use the published model on HuggingFace, 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 (cfr. Section 4.5).

⁵ The entities are retrieved from a small candidates set, considering that entity vocabulary could be quite large.

⁶ 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 pre-train the selected approaches.

Approach	Dataset	Entities	Source	Entities domain
Alligator [9]	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 [20]	TURL WikiTable w/o WikiGS [93]	1.23M	Web Tables	Cross
TableLlama [22]	TableInstruct [22]	2.6M	Web Tables	Cross

4. Distribution-aware evaluation of performance, generalisation, and efficiency

The main objective of this work is to compare GTUM and feature-based ML approaches on the CEA task within a well-defined and common evaluation setting. As specified by the SemTab challenge [25], the metrics adopted to evaluate an approach are:

$$\text{Precision}(P) = \frac{\# \text{ correct annotations}}{\# \text{ submitted annotations}} \quad (1)$$

$$\text{Recall}(R) = \frac{\# \text{ correct annotations}}{\# \text{ groundtruth annotations}} \quad (2)$$

$$\text{F1} = \frac{2PR}{P + R} \quad (3)$$

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 [20,22]. Under this setting, Precision, Recall and F1 are the same and equal to the Accuracy metric. Therefore, we report the accuracy as:

$$\text{Accuracy} = \frac{\# \text{ correct annotations}}{\# \text{ mentions to annotate}} \quad (4)$$

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 testing 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 DAGOBAN (with some limitations) on the datasets used to 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 4.4) and the training of the models (Section 4.5).

4.1. Datasets

The datasets considered in this work come from different sources and contain information about different domains. All datasets target Wikidata as the reference KG; candidate entities and ground-truth annotations are therefore expressed as publicly available Wikidata identifiers. In particular, we have selected the following:

- **SemTab2021 - R3 (BioDiv)** [94]: 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)** [95]: 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 (HT))** [96]: datasets with tables generated using SPARQL queries [10]. The datasets used from HardTables 2022 are *round 1* (R1) and *round 2* (R2);
- **SemTab2023 - R1 (WikidataTables)** [25]: 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 not being provided;
- **TURL** [20]: the authors of TURL developed the TURL dataset 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 3 reports the statistics of each dataset used for testing and fine-tuning the selected approaches († indicates the datasets that have undergone a reduction). 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*.

Table 3
Statistics of the datasets used to fine-tune and evaluate models.

Dataset	Split	Tables	Cols (min max $\bar{x}$)	Rows (min max $\bar{x}$)	Entities
SemTab2022 2T-red	Train ^a	91	1 8 4,86	5 369 98,61	14 674
	Test ^a	26	1 8 4,65	7 264 74,58	4 691
SemTab2022 HT-R1	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
SemTab2022 HT-R2	Train ^a	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 WikidataTables-R1	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
SemTab2021 BioDiv-red	Test ^a	11	1 26 17,45	26 100 58,90	1 232
TURL-120k	Train	13 061	1 43 5,43	1 624 12,98	120 000
TURL-2k-red	Test	1 295	1 14 1,03	6 257 32,95	1 801

^a 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.

4.2. Prompt specification and dataset adaptation

Prompts follow the “Entity Linking” template from [22] and include the following segments: an *Instruction* that provides context and task guidelines; an *Input* table, Markdown-serialised and a *Question* asking which of the provided referent candidates corresponds to a specific table mention. The *Generated* segment contains the model’s predicted answer. To support understanding, a minimal, self-contained illustrative example of the prompt, candidate list, and expected answer format is provided in Fig. 2. 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 DAGOBAB were created using the same mentions as TableLlama.

We also verified that there is no direct overlap between the rows used for training and those used for testing in our experimental setting: train/test splits follow the original SemTab and TURL protocols, and our reductions (*BioDiv-red*, *2T-red*, *TURL-2K-red*) only remove rows to satisfy TableLlama’s context-length constraints without introducing additional leakage.

4.3. 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 [10,20,25,94,95]. 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 [22]. 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 [20], 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.

Table 2 links each model to the datasets used in its main training phase. Combining this information with the test datasets in Table 4 clarifies why a given dataset is considered IN, MOOD, or OOD for each approach, and it helps interpret the generalisation patterns reported in Section 5.1.

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 and detailed in Table 3, 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.

Entity Linking Prompt Example

Instruction: *This is an entity linking task. The goal for this task is to link the selected entity mention in the table cells to the entity in the knowledge base. You will be given a list of referent entities, with each one composed of an entity name, its description and its type. Please choose the correct one from the referent entity candidates.*

Input: [TLE] List of high schools in South Dakota.

```
col: |school|type|city|county|mascot|
row 0: |Aberdeen High School|Private|Aberdeen|Brown|Knights|
row 1: |Agar High School|Public|Agar|Sully|
[...]
row 35: |Crazy Horse High School|Public|Wanblee|Jackson|Chiefs|
row 36: |Crow Creek High School|Public|Stephan|Hyde|Chieftains|
row 37: |Custer High School|Public|Custer|Custer|Wildcats|
[...]
```

Question: *The selected entity mention is: ‘Hyde’. The column name for ‘Hyde’ is ‘county’. The referent candidates are:*

```
<Dr. Jekyll and Mr. Hyde [DESC] fictional characters [TYPE] group of
fictional characters>
<Hyde v Hyde [DESC] landmark case of the English Court of Probate and
Divorce [TYPE] legal case>
<Hyde County [DESC] county in South Dakota, United States [TYPE] county
of South Dakota>
<Douglas Hyde [DESC] first President of Ireland (1860-1949) [TYPE]
linguist>
<Hyde County [DESC] county in North Carolina, United States [TYPE]
county of North Carolina>
<Hyde [DESC] civil parish in Bedfordshire, UK [TYPE] civil parish>
<Hyde Park [DESC] town in Dutchess County, New York, United States
[TYPE] town of New York>
<Strange Case of Dr Jekyll and Mr Hyde [DESC] novel by R. L. Stevenson
[TYPE] literary work>
<Hyde Park [DESC] neighborhood in Chicago, Illinois [TYPE] neighborhood>
[...]
```

What is the correct referent entity for ‘Hyde’?

Answer: <Hyde County [DESC] county in South Dakota, United States
[TYPE] county of South Dakota>

Fig. 2. The prompt structure for GPT-4o, GPT-4o-mini and TableLlama, following the “Entity Linking” template from [22].

2. We fine-tune TURL, TableLlama and Alligator on the train splits from their MOOD data, i.e., on SemTab2022 HT-R1, SemTab2022 HT-R2, SemTab2023 WikidataTables-R1 and SemTab2022 2T-red for TURL and TableLlama, and on TURL120k⁷ for Alligator; finally we test the fine-tuned models on our test data.

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;

⁷ TURL-120k has been created starting from the original TURL dataset [20] by sub-sampling 13061 tables containing 120000 mentions (≈ 125000 is the number of mentions used to fine-tune TableLlama and TURL on their MOOD data combined).

Table 4

Test datasets denomination, as specified in Section 4.3. For every dataset, we report its source, the domain of the entities contained, and the characterisation as In-Domain (IN), Out-Of-Domain (OOD), and Moderately-Out-Of-Domain (MOOD) for each of the pre-trained models (see Table 2 for the pre-training datasets). {Dataset}-red means that the specific dataset has been reduced as explained in Section 4.1.

Dataset	Source	Entities domain	Approaches		
			Alligator	TURL	TableLlama
SemTab2022 2T-red	Graph Queries/Web Tables	Cross	IN	MOOD	MOOD
SemTab2022 HT-R1	Graph Queries	Cross	IN	MOOD	MOOD
SemTab2022 HT-R2	Graph Queries	Cross	IN	MOOD	MOOD
SemTab2023 WikidataTables-R1	Graph Queries	Cross	IN	MOOD	MOOD
SemTab2021 BioDiv-red	Biodiversity Tables	Specific	OOD	OOD	OOD
TURL-120k	Web Tables	Cross	MOOD	IN	IN
TURL-2K-red	Web Tables	Cross	MOOD	IN	IN

- Compare the approaches in inference time and occupied memory to get insights about their applicability on application domains where the 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.

4.4. Candidate entity retrieval with LamAPI

The candidates for the mentions contained in the TURL dataset [20], along with its sub-sampled version TURL-2K [22], were retrieved through the Wikidata-Lookup-Service,⁸ which is known to have a low coverage w.r.t. other ERs [97]. For this reason, we researched to identify a state-of-the-art approach 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. In *LamAPI*, an ElasticSearch⁹ index of the Wikidata dump (v. 01072024) has been constructed, leveraging an engine designed to search and analyse extensive data volumes in nearly real-time swiftly. Accordingly, all entity disambiguation experiments in this paper target Wikidata as the sole reference KG; candidate entities, ground-truth annotations, and evaluation metrics are therefore defined with respect to publicly available Wikidata identifiers. 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) [97].

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 ($\approx 48\%$) 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 ($\approx 91\%$) 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 the TURL-2K-red dataset, which we called *TURL-2K-red-LamAPI*.

4.5. Training and implementation details

Models Pre-training. We first replicated the TURL pre-training with the same hyperparameters as specified by the authors in [20] but in a distributed setting on 4 80 GB-A100 GPUs, following the findings in [98], 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.¹⁰ Alligator is pre-trained on different SemTab datasets before 2022 as in the original paper.¹¹ We refer to Table 2 for details about the datasets used for pre-train. DAGOBAN is run with the default hyperparameters as specified in the corresponding GitHub repository.¹²

MOOD Fine-tuning. We remind that we consider two evaluation settings, given the same test datasets (cfr. Table 3): (1) the approaches are tested based on their pre-trained state; (2) TURL, TableLlama and Alligator are first fine-tuned on their MOOD data (cfr. Tables 3 and 4) and subsequently tested. For the fine-tuning of TURL the default hyperparameters have been adopted as specified by the authors in [20]. For TableLlama we have employed the findings in [99], i.e., rewarming the learning rate from $\eta_0 = 0.0$ to $\eta_{max} = 2e-5$ for 0.5% of the training iterations, then redecaying it with a cosine scheduler to reach $\eta_{min} = 0.1 \cdot \eta_{max}$

⁸ <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”.

⁹ <https://www.elastic.co/>.

¹⁰ <https://huggingface.co/oslup/TableLlama>.

¹¹ As reported in Table 2, 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.

¹² <https://github.com/Orange-OpenSource/Table-Annotation>.

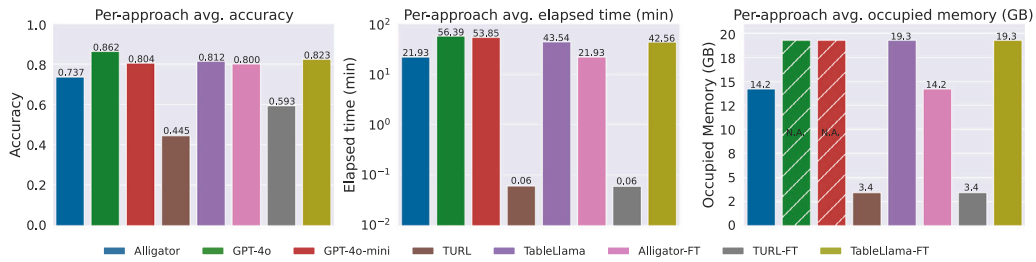


Fig. 3. Per-approach average accuracy, elapsed time (min), and occupied memory (GB). The time reported is the elapsed time to link a mention, without considering the ER time. The occupied memory is the occupied GPU memory for TURL and TableLlama, while is the “Virtual Memory Size” for Alligator and DAGOBah. {Approach}-FT represents the respective approach fine-tuned in MOOD setting (Section 4.3). “O.O.M”. stands for Out-Of-Memory.

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 [100] following Chen et al. [28] with a micro batch-size= 1, 64 gradients accumulation steps, LoRA-rank = 8, LoRA- α = 16, without any dropout or weight-decay. Finally, we have fine-tuned the Alligator’s second model (RN) with the default hyperparameters specified in [9].

Evaluation and inference. Following our primary objective, all selected approaches were evaluated on the same test data (refer to Table 3). Both TableLlama and TURL were run with a batch size of 1, though this setting has distinct interpretations: for TURL, it means processing one table at a time, while for TableLlama, it means processing a single mention at a time. The GPT-4o and GPT-4o-mini models were also run with TableLlama-style prompts, meaning they process one mention at a time. In contrast, Alligator was run with its default configuration, using 8 parallel processes, each handling up to 50 rows per table. Lastly, DAGOBah was executed with its default hyperparameters, as specified in its official repository.¹³

Technical infrastructure. Both test and fine-tuning for TableLlama and TURL has run on the Azure NC24ads_a100_v4; Alligator on an Intel(R) Xeon(R) Gold 6238R CPU @ 2.20 GHz with 64 cores and 96 GB of RAM, while DAGOBah on an Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00 GHz with 32 cores and 96 GB of RAM.

5. Comparative results and discussion

We first focus on the main results, and then we discuss evidence from ablation studies. We distinguish three dimensions of efficiency: (i) execution time, measured as the average wall-clock time needed to link a mention (excluding entity retrieval); (ii) memory usage, measured as peak GPU memory for TURL and TableLlama and virtual memory size for Alligator and DAGOBah; and (iii) monetary cost, expressed in USD and estimated from Azure VM prices and OpenAI API prices (cfr. footnote 16).

5.1. Main results

Fig. 3 reports the per-approach average accuracy, with the average computed over the test datasets (cfr. Table 3), while Table 5 reports the detailed performances achieved by the four different approaches on the test data. 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 pattern reflects the different inductive biases discussed in Section 3. By design, Alligator relies on explicit KG structure and lexical/syntactic similarity (e.g., entity popularity, token overlap, row-level literal matching, column-type and property consistency). Such a bias is particularly well aligned with SemTab-style benchmarks, where tables are constructed from KGs and their structure closely follows the underlying graph. In contrast, TURL and TableLlama are GTUM models that are trained on web tables and are therefore biased towards well-defined tables structure, metadata, and high-level semantic patterns in the textual content. This makes them more robust on *BioDiv-red* and *TURL-2K-red-LamAPI*, where the link to the KG is weaker and the textual description is richer.

Moreover, the datasets they excel on are the ones considered *in-domain* (IN). Surprisingly, both TURL and TableLlama excel on BioDiv, even though it is considered OOD for both approaches: we hypothesise that the enormous and general pre-training knowledge these models retain 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 (cfr. the ablation study in Section 5.2). Regarding the GPT family, GPT-4o is, on average, the best-performing model, with GPT-4o-mini that is on par with the MOOD fine-tuned version of Alligator.

¹³ <https://github.com/Orange-OpenSource/Table-Annotation/tree/main>.

Table 5

Performance of the four algorithms on our test data, denoted as “Accuracy^{Data Typology}”, where the superscripts represent the data typology: ^{IN} *in-domain*, ^{MOOD} *moderately out-of-domain*, ^{OOD} *out-of-domain*, and ^{MOOD→IN} indicates the Data Typology transition from MOOD to IN after the model has been fine-tuned in MOOD setting. The best model is highlighted in **bold**, while the second-best model is underlined. {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	TURL	TableLlama	Alligator	DAGOBAB	GPT-4o	GPT-4o-mini	TURL-FT	TableLlama-FT	Alligator-FT
SemTab2022 2T-red	0.1343 ^{MOOD}	0.8243 ^{MOOD}	0.7156 ^{IN}	N.A. ^{IN}	0.8783 ^{UNK}	0.7907 ^{UNK}	0.3323 ^{MOOD→IN}	0.8400 ^{MOOD→IN}	<u>0.8531</u> ^{IN}
SemTab2022 HT-R1	0.3997 ^{MOOD}	0.7873 ^{MOOD}	0.8890 ^{IN}	0.7413 ^{IN}	<u>0.8542</u> ^{UNK}	0.8087 ^{UNK}	0.7454 ^{MOOD→IN}	0.8001 ^{MOOD→IN}	0.8165 ^{IN}
SemTab2022 HT-R2	0.2763 ^{MOOD}	0.6619 ^{MOOD}	0.8218 ^{IN}	0.6289 ^{IN}	<u>0.7516</u> ^{UNK}	0.6565 ^{UNK}	0.6018 ^{MOOD→IN}	0.6778 ^{MOOD→IN}	0.7212 ^{IN}
SemTab2023 WikidataTables-R1	0.3391 ^{MOOD}	0.7426 ^{MOOD}	0.8248 ^{IN}	0.7279 ^{IN}	<u>0.8034</u> ^{UNK}	0.7554 ^{UNK}	0.7061 ^{MOOD→IN}	0.7530 ^{MOOD→IN}	<u>0.8086</u> ^{IN}
SemTab2021 BioDiv-red	0.8109 ^{OOD}	0.9513 ^{OOD}	0.5674 ^{OOD}	N.A. ^{OOD}	0.9748 ^{UNK}	0.9497 ^{UNK}	0.6347 ^{OOD}	<u>0.9610</u> ^{OOD}	0.8547 ^{OOD}
TURL-2K-red-LamAPI	0.7118 ^{IN}	<u>0.9051</u> ^{IN}	0.6017 ^{MOOD}	O.O.M. ^{MOOD}	0.9095 ^{UNK}	0.8629 ^{UNK}	0.5347 ^{IN}	0.9045 ^{IN}	0.7456 ^{MOOD→IN}

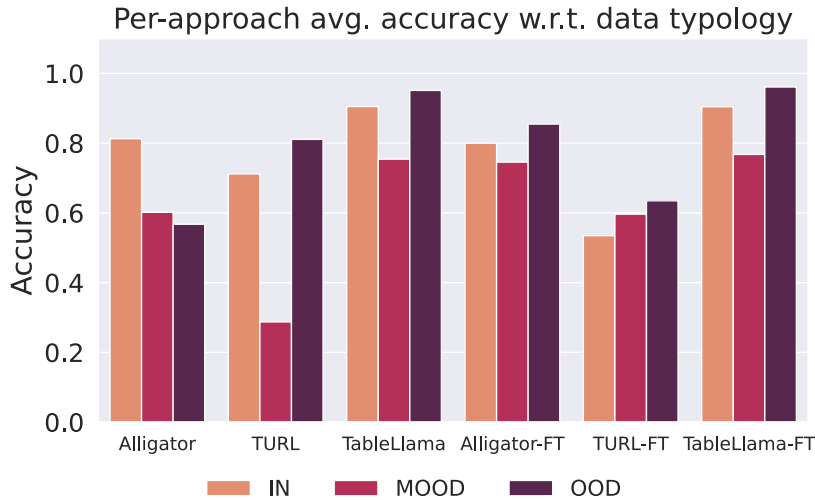


Fig. 4. Per-approach accuracy concerning data typology, as defined in Section 4.3. The average is computed over datasets sharing the same data typology.

MOOD fine-tuning. Table 5 also reports the performances of TURL, TableLlama and Alligator after MOOD fine-tuning. Thanks to the “continual finetuning” inspired by Ibrahim et al. [99] 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 ($\approx +36\%$ on WikidataTables-R1) but suffers a severe drop in performance on both IN and OOD data, confirming the catastrophic forgetting suffered by models after fine-tuning [101]. In particular, Table 5 shows that TURL-FT improves substantially on SemTab2022 2T-red, HT-R1, HT-R2, and SemTab2023 WikidataTables-R1, but its accuracy drops on *BioDiv-red* and *TURL-2K-red-LamAPI*. This suggests that updating all model parameters on SemTab-like data biases TURL towards KG-generated tables and away from the web tables it was originally trained on, forgetting part of the knowledge acquired from the WikiTables pre-training. Alligator achieves great performances on 2T-red, *BioDiv-red* and *TURL-2K-red-LamAPI* after MOOD fine-tuning, while suffering a contained drop in performance (apart from HT-R2, where it loses 10%), demonstrating a better generalisation pattern than TURL. The average performance concerning the data typology is depicted in Fig. 4, which further demonstrates these findings.

Computational performance. Fig. 3 also reports the average elapsed time (center) for the six tested approaches and the average occupied memory (right) by each approach (TURL and TableLlama’s occupied memory refers to the GPU memory, while the Alligator and DAGOBAB’s memory refers to the overall “Virtual Memory Size”). Note that time is on a logarithmic scale and is considered as the time employed by an approach to link a mention without considering the ER time.¹⁴ 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. execution time and occupied memory. DAGOBAB, on the other hand, is completely out-of-bounds, especially if one is bounded by resource or budget constraints.

¹⁴ The GPT-4o and GPT-4o-mini time is constrained by the limits imposed by [OpenAI API rate limits](#).

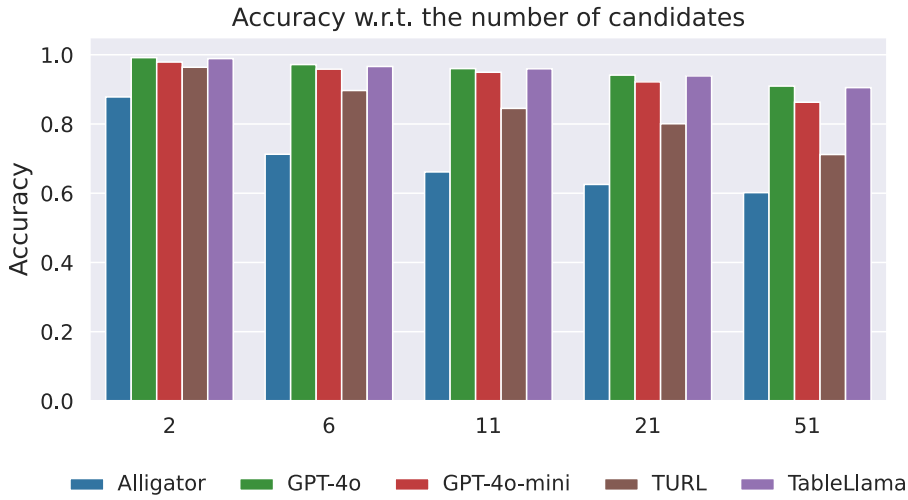


Fig. 5. Accuracies achieved by Alligator, GPT-4o, TableLlama and TURL w.r.t. the number of candidates on the *TURL-2K-red-LamAPI* dataset.

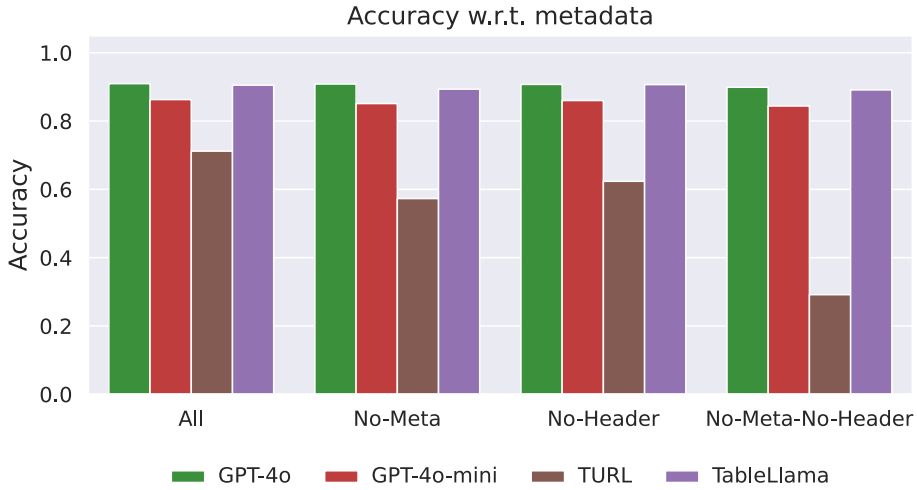


Fig. 6. Accuracies achieved by GPT-4o, TableLlama and TURL w.r.t. the presence of table's metadata on the *TURL-2K-red-LamAPI* dataset.

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. Fig. 5 reports the accuracy of Alligator, TURL, TableLlama, GPT-4o and GPT-4o-mini given a different number of candidates per mention, also considering 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 TURL, TableLlama, GPT-4o, and GPT-4o-mini, testing on the *TURL-2K-red-LamAPI* dataset. We ran experiments 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 $[\text{col}0, \text{col}1, \dots, \text{col}N]$ and (iii) *No-Meta-No-Header*, i.e., the combination of both (i) and (ii). From Fig. 6, 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 and GPT-4o are 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.

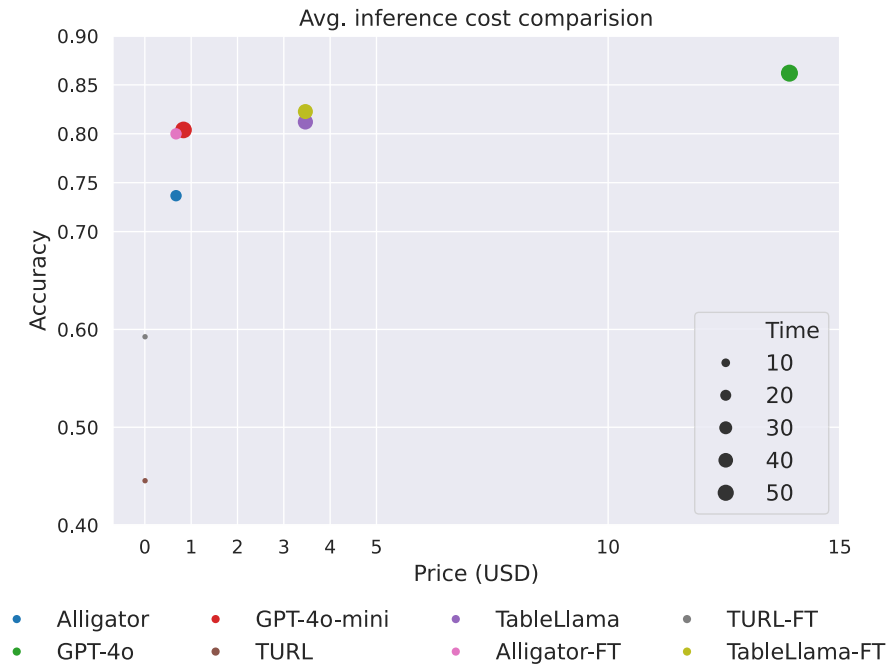


Fig. 7. Per-approach accuracy over price, on average. The price is computed given the average time spent (in minutes) to run the evaluation over all the test data.

5.3. Discussion

Generative GTUM approaches with a high number of parameters 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, as found in Sections 5.1 and 5.2; however, specific STI approaches trained on in-domain data, and optionally fine-tuned on moderately-out-of-domain data, using a tiny fraction of these parameters can still be on par with generalistic approaches while being faster and occupying less memory. STI-approaches have been designed as table-to-KG matching approaches and consider different values associated with the candidate entities in KG in the signals they use for disambiguation. As a result, they are still advantageous when the values in the table match entity properties in the KG. Otherwise, GTUM and other LLM-based approaches only consider candidate names, types and descriptions; it is likely that prior parametric knowledge about entities also plays a role in their performance. On web tables, this information seems somehow sufficient to outperform STI-based approaches. Consequently, when tables are not likely to contain values matching in the KGs these approaches seem more promising. Moreover, approaches like GPT-4o and TableLlama are 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 efficiency, 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 models of the size of TableLlama has still enormous monetary costs (48 A100-80 GB for 9 training days), so devising generative methods based on smaller LLMs, e.g., Phi [102], could be an interesting research direction. However, more sophisticated approaches may be needed to achieve the same level of generalisation and reliability.

Moreover, Fig. 7 shows the accuracy achieved by the tested approaches, both before and after MOOD fine-tuning, with respect to the average price¹⁵ paid to host machines to run inference on all the test data. The figure shows how GPT-4o while being the best-performing model on average, is also the most expensive one; GPT-4o-mini, TableLlama, TableLlama-FT, and Alligator-FT are comparable in terms of accuracy and price paid, although TableLlama costs are slightly higher.

These results suggest that generative GTUM approaches are best suited when the tables to be annotated contain rich, highly semantic textual data. GPT-4o is the most accurate among these methods but also the most expensive. GPT-4o-mini offers a fair compromise between accuracy, monetary cost, and computational demand. On the other hand, Alligator is more appropriate when

¹⁵ For TableLlama, TURL and Alligator, prices are estimated using the [Azure Pricing Calculator](#), selecting the [NC24ads_a100_v4](#) for running GPU inference with TableLlama and TURL, and [D48plds_v5](#) for running CPU inference and hosting Alligator. GPT-4o and GPT-4o-mini prices are estimated from the [OpenAI prices page](#), considering a token as being equal to 4 characters.

Table 6

Comparative overview of TURL, TableLlama, and Alligator, highlighting recommended use cases, primary advantages, and notable limitations for selecting appropriate entity disambiguation approaches in different contexts.

Approach	Recommended use cases	Key advantages	Notable limitations
TURL	• Small-scale or web-based tables with consistent metadata • Environments with limited GPU resources	• Lightweight architecture enabling fast inference • Straightforward deployment in resource-constrained environments	• Performance decreases with many candidate entities • Heavily relies on metadata for context • No support for NIL detection
TableLlama	• Complex or diverse-domain tables requiring deep semantic understanding • Environments with sufficient GPU resources	• Strong generalisation on out-of-domain data • Large context window (up to 8192 tokens), accommodating complex table structures	• High computational and memory demands • Limited by fixed context length for extremely large tables • No integrated NIL-awareness or confidence estimation
Alligator	• Tables closely aligned with Knowledge Graphs (KGs) and benefiting from property-based matching • Scenarios requiring CPU-efficient and parallelizable methods • Linking tasks that benefit from confidence scores	• Direct utilisation of KG properties for accurate disambiguation • Provides explicit confidence estimations • Does not require GPU resources	• Less effective with purely textual or semantic-based mentions • Often requires domain-specific fine-tuning • Limited semantic understanding compared to larger LLMs

signals from the knowledge graph (KG) significantly impact the final accuracy, especially when time and budget are limited. When fine-tuned, Alligator performs comparably to GPT-4o-mini and TableLlama in terms of accuracy with a reduced cost. Finally, TableLlama and TURL are not NIL aware and TableLlama and GPT-4o, 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.

5.4. Qualitative error analysis

To better understand the limitations of the different approaches, we conducted a qualitative inspection of error cases on the SemTab2022 HTR1 dataset. We observed that generative models like GPT-4o excel at inferring the latent domain of a table, often leveraging pre-training knowledge to resolve ambiguous mentions by identifying coherence across a row. However, this reasoning is not always robust; the model occasionally exhibits inconsistencies, correctly identifying the domain for most rows while inexplicably misclassifying others in the same context. For instance, in a table listing rolling stock (e.g., “ED4”, “BOA”), GPT-4o correctly identified the domain for most entries but linked “ED9” to a biological cell line, ignoring the clear column-wise context. Furthermore, generative models show a tendency towards “granularity drift”, often linking to broader or more popular entities—such as annotating “Alba Iulia” as the municipality rather than the specific constituent locality required by the ground truth—and occasionally hallucinating specific details (e.g., assigning a specific year to “Hurricane Paul” in an underspecified table) to resolve mentions. In contrast, the feature-based Alligator avoids generative hallucinations by strictly adhering to the candidate list. However, it is susceptible to “semantic drift” when local features, such as string similarity or global popularity, outweigh the semantic cues provided by the row context. For example, it may link a transportation code to a popular commercial vehicle rather than the correct local transit line, failing to leverage the geospatial context that a generative model might implicitly have. Finally, all approaches falter when tables lack distinguishing information entirely, though they fail differently: generative models tend to invent the missing context, while feature-based models default to the statistically most probable candidate.

6. Implications for semantics management and semantics-driven system engineering

Table 6 synthesises the recommended use cases, key advantages, and notable limitations of the approaches evaluated in this study. Beyond providing practitioners with actionable guidance for selecting the most suitable entity disambiguation method, the table and the evidence behind it carry broader implications for semantic data management and semantics-driven systems engineering.

A central finding of our study is that the semantic overlap between the data in the table and the data in the KG should drive the choice of disambiguation approach. Overlapping values and structure (e.g., column pairs with semantics and values shared with properties in the KG) help build useful signals in the extracted features, making type and property-related features more helpful for disambiguation. In this case, approaches based on table-to-graph matching features like Alligator offer a favourable trade-off between accuracy, cost, and interpretability, making them well-suited for large-scale KG construction and maintenance pipelines. Conversely, when tables originate from sources that have a looser structure, few columns, and different text values (e.g., open web tables, scientific publications, or business spreadsheets), engineered features provide weak signals and the semantic mediation must rely more on matching based on implicit, language-based, and context-sensitive signals, better captured by LLMs. In these cases (examples of which can be found also in [15,16]), generative LLM-based approaches such as TableLlama or GPT-4o prove more

robust. In both cases, the choice of method shapes the degree to which formal KG semantics vs. parametric knowledge guides the annotation, a distinction that is central to the design of semantics-driven information systems (cfr. Fig. 1).

From an engineering perspective, these findings suggest a modular architecture for semantics-driven tabular data processing: a disambiguation module grounded in table-to-KG matching features can be complemented by an LLM-based module that handles mentions for which KG signals are weak or absent. Such a hybrid design would align with the established vision in semantic-driven system engineering, where explicit semantic representations (KGs) are combined with context-sensitive and language-based operations (LLMs), enabling systems that are both methodologically grounded and adaptable to evolving data landscapes. Our evaluation framework, with its distribution-aware IN/MOOD/OOD protocol and multi-dimensional assessment of accuracy, efficiency, and cost, offers a reusable methodology for benchmarking new disambiguation components as they are integrated into such architectures.

The *MammoTab 25* benchmark [92] provides complementary evidence that reinforces these conclusions. Among the 19 open-weight LLMs evaluated on a curated subset of 870 tables (approximately 85 000 cells), TableLlama (7B, fine-tuned) achieves an overall accuracy of approximately 73%, nearly matching the best general-purpose model, Mistral-Large-123b-it (123B parameters), at approximately 74%. The best general-purpose 7B model, Qwen2.5-7B, reaches only about 51%, confirming that a generalist model requires roughly an order-of-magnitude increase in parameters to approach the accuracy of a fine-tuned 7B model. This corroborates the competitiveness of TableLlama observed in our experiments and suggests that fine-tuning smaller, more capable base models than Llama 2, on which TableLlama is based, is a promising and cost-effective research direction. Notably, the *MammoTab 25* evaluation focuses exclusively on open-weight models and does not include proprietary LLMs such as GPT-4o; our study complements it by showing that GPT-4o still achieves the highest accuracy overall, but at a significantly higher monetary cost (cfr. Fig. 7).

From a sustainability perspective [103], the cost of training and running large generative models is a concrete concern for practical STI deployments: TableLlama alone required 48 NVIDIA A100-80 GB GPUs for nine days of training. Scaling to even larger models would further increase energy consumption and carbon footprint. Lightweight approaches such as Alligator, which runs on CPU, and compact fine-tuned models therefore retain a clear practical advantage in resource-constrained or sustainability-conscious settings, an aspect that should be factored into semantic-driven system engineering design decisions.

A further challenge for the adoption of LLM-based disambiguation in practical applications is the *absence of native confidence signals*. Neither TableLlama nor GPT-4o produce per-annotation confidence scores, which limits their applicability in human-in-the-loop (HITL) validation workflows, a common requirement in real-world semantic data management. In contrast, Alligator provides explicit confidence estimates for each candidate, enabling practitioners to triage uncertain annotations for manual review. The importance of such signals is illustrated by recent work on building canonical registers of public-sector entities through entity linking on structured data [15], where the adoption of LLM-based approaches was hindered precisely by the lack of reliable confidence estimates. A first approach to address this gap has been recently proposed by Bono et al. [104], who introduce an efficient method to estimate the uncertainty of LLM-based entity linking predictions; their results show that acceptable confidence estimates can be obtained, especially for fine-tuned models such as TableLlama, opening a promising direction for integrating generative STI models into HITL pipelines.

Finally, the predominance of decoder-only architectures in recent STI research has left encoder-based approaches largely unexplored with modern pre-trained models. TURL, the encoder-only model evaluated in this study, still relies on TinyBERT, a compact encoder released in 2020. Retraining a TURL-like architecture with newer, more powerful encoders could offer a favourable accuracy/efficiency trade-off: encoder models are inherently faster at inference, produce contextualised cell embeddings amenable to confidence estimation, and can be pre-trained on tabular corpora at a fraction of the cost of large decoder-only LLMs. Exploring this direction would complement the generative paradigm and broaden the design space available to engineers of semantics-driven systems.

7. Conclusions and future works

This study addressed the lack of comprehensive comparative analyses among different approaches to STI, focusing specifically on the CEA task. We evaluated four STI approaches, Alligator, TURL, TableLlama, and DAGOBAB (the latter with a partial evaluation due to its high computational demands), as well as two proprietary LLMs (GPT-4o and GPT-4o-mini), all under common conditions. Our evaluation spanned multiple dimensions: accuracy, generalisation, execution time, memory usage, and cost. We tested these methods across various scenarios, i.e., “in-domain”, “out-of-domain” and “moderately out-of-domain”. Our experiments suggest that approaches based on a large generative LLM excel in accuracy and generalisation, as demonstrated by the results on MOOD and OOD data (cfr. Table 5 and Fig. 4), at the price of an increased execution and training time. TURL, an encoder-only model based on TinyBERT, is the most efficient but lacks generalisation capabilities. Moreover, our experiments suggest that specific STI models like Alligator, despite their limited number of parameters, can still be a viable option when resources and budget are limited, with performance improvements obtained after fine-tuning. Finally, GPT-4o, while being the most accurate, is also the most expensive, with GPT-4o-mini representing a better trade-off regarding accuracy and cost.

Future works include the possibility to train a smaller and cheaper LLM-based table model, e.g., Phi [102], while enabling also the handling of NIL entities and a score associated with cell annotations. Additionally, we aim to augment LLM-based approaches with advanced prompting techniques like In-Context Learning or Chain-of-Thought reasoning. This enhancement could leverage the inherent reasoning capabilities of LLMs and facilitate human-in-the-loop interactions to guide the model’s behaviour. Another potential direction is to modify models like TURL and TableLlama to operate at the cell level rather than the table level. This change could reduce memory consumption and allow for the application of standard data augmentation techniques, potentially improving both performance and efficiency.

CRediT authorship contribution statement

Federico Belotti: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization. **Marco Cremaschi:** Writing – review & editing, Writing – original draft, Supervision. **Fabio Dadda:** Writing – review & editing, Writing – original draft, Software, Data curation. **Roberto Avogadro:** Writing – review & editing, Software. **Matteo Palmonari:** Writing – review & editing, Writing – original draft, Supervision, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Federico Belotti reports financial support was provided by enRichMyData (HE 101070284). Matteo Palmonari reports financial support was provided by enRichMyData (HE 101070284). Federico Belotti reports financial support was provided by Discount Quality for Responsible Data Science (202248FWFS). Matteo Palmonari reports financial support was provided by Discount Quality for Responsible Data Science (202248FWFS). Roberto Avogadro reports a relationship with SINTEF Oslo that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been partially funded by the European innovation actions enRichMyData (HE 101070284) and DataPACT (HE 101189771), and the Italian PRIN project Discount Quality for Responsible Data Science: Human-in-the-Loop for Quality Data (202248FWFS) funded by the European Community - Next Generation EU.

Data availability

The data used in this study is publicly available on HuggingFace at <https://huggingface.co/datasets/belerico/evaluating-llms-on-tabular-data>.

References

- [1] R. Buchmann, J. Eder, H.-G. Fill, U. Frank, D. Karagiannis, E. Laurenzi, J. Mylopoulos, D. Plexousakis, M.Y. Santos, Large language models: Expectations for semantics-driven systems engineering, *Data Knowl. Eng.* 152 (2024) 102324, <http://dx.doi.org/10.1016/j.datak.2024.102324>, URL: <https://www.sciencedirect.com/science/article/pii/S01690223X2400048X>.
- [2] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, *Commun. ACM* 57 (10) (2014) 78–85, <http://dx.doi.org/10.1145/2629489>, URL: <https://doi.org/10.1145/2629489>.
- [3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z.G. Ives, DBpedia: A nucleus for a web of open data, in: ISWC/ASWC, 2007, URL: <https://api.semanticscholar.org/CorpusID:7278297>.
- [4] N. Abdelmageed, E. Jiménez-Ruiz, O. Hassanzadeh, B. König-Ries, KG2Tables: A Domain-Specific Tabular Data Generator to Evaluate Semantic Table Interpretation Systems, *Trans. Graph Data Knowl.* 3 (1) (2025) 1:1–1:28, <http://dx.doi.org/10.4230/TGDK.3.1.1>, URL: <https://drops.dagstuhl.de/entities/document/10.4230/TGDK.3.1.1>.
- [5] J. Liu, Y. Chabot, R. Troncy, V.-P. Huynh, T. Labbé, P. Monnin, From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods, *J. Web Semant.* 76 (2023) 100761.
- [6] M. Pham, S. Alse, C.A. Knoblock, P. Szekely, 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, Springer, 2016, pp. 446–462.
- [7] Y. Suhara, J. Li, Y. Li, D. Zhang, Ç. Demiralp, C. Chen, W.-C. Tan, Annotating columns with pre-trained language models, in: *Proceedings of the 2022 International Conference on Management of Data, SIGMOD '22, Association for Computing Machinery, New York, NY, USA, 2022*, pp. 1493–1503.
- [8] D. Rao, P. McNamee, M. Dredze, Entity linking: Finding extracted entities in a knowledge base, in: *Multi-Source, Multilingual Information Extraction and Summarization*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 93–115.
- [9] R. Avogadro, M. Ciavotta, F. De Paoli, M. Palmonari, D. Roman, 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, 2023*, pp. 142–149.
- [10] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, SemTab 2019: Resources to benchmark tabular data to knowledge graph matching systems, in: *The Semantic Web*, Springer, Cham, 2020, pp. 514–530.
- [11] M. Cremaschi, B. Spahiu, M. Palmonari, E. Jimenez-Ruiz, Survey on semantic interpretation of tabular data: Challenges and directions, 2024, arXiv preprint [arXiv:2411.11891](https://arxiv.org/abs/2411.11891).
- [12] M. Ciavotta, V. Cutrona, F. De Paoli, N. Nikolov, M. Palmonari, D. Roman, Supporting semantic data enrichment at scale, in: *Technologies and Applications for Big Data Value*, Springer, 2022, pp. 19–39.
- [13] F. De Paoli, M. Ciavotta, R. Avogadro, E. Hristov, M. Borukova, D. Petrova-Antonova, I. Krasteva, An interactive approach to semantic enrichment with geospatial data, *Data Knowl. Eng.* 153 (2024) 102341, <http://dx.doi.org/10.1016/j.datak.2024.102341>.
- [14] V. Cutrona, F. De Paoli, A. Košmerlj, N. Nikolov, M. Palmonari, F. Perales, D. Roman, Semantically-enabled optimization of digital marketing campaigns, in: *ISWC*, Springer, 2019, pp. 345–362.
- [15] R. Avogadro, I. Makgill, A. Thomas, A. Soylu, D. Roman, Building a canonical register of public sector entities: Semantic linking of procurement data at scale, in: *CEUR Workshop Proceedings*, Vol. 4085, 2025, pp. 5–8, URL: <https://ceur-ws.org/Vol-4085/paper2.pdf>.
- [16] I. Novalija, D. Roman, F. Belotti, V. Alexiev, L. Rei, R. Avogadro, B. Khalilvandian, B. Bechev, C. Alexandra Chinie, I. Ciurea, J. Brank, C. Udriou, A. Soylu, M. Palmonari, From code to concept: A semantic approach to AI innovation discovery in open source software repositories, *IEEE Access* 13 (2025) 129995–130014, <http://dx.doi.org/10.1109/ACCESS.2025.3590135>.
- [17] B. Steenwinckel, F. De Turck, F. Ongena, MAGIC: Mining an augmented graph using INK, starting from a CSV., in: *SemTab@ ISWC*, 2021, pp. 68–78.
- [18] C. Sarthou-Camy, G. Jourdain, Y. Chabot, P. Monnin, F. Deuzé, V.-P. Huynh, J. Liu, T. Labbé, R. Troncy, DAGOBAB-UI: a new hope for semantic table interpretation, in: *ESWC Demo Papers*, Springer, 2022, pp. 107–111.

- [19] A. Alidu, M. Ciavotta, F. De Paoli, SemT: A framework for enhancing tabular data through enrichment-as-a-service, in: *European Conference on Service-Oriented and Cloud Computing*, Springer, 2025, pp. 33–39.
- [20] X. Deng, H. Sun, A. Lees, Y. Wu, C. Yu, Turl: Table understanding through representation learning, *ACM SIGMOD Rec.* 51 (1) (2022) 33–40, <http://dx.doi.org/10.1145/3542700.3542709>.
- [21] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (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)*, Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186.
- [22] T. Zhang, X. Yue, Y. Li, H. Sun, TableLLama: Towards open large generalist models for tables, 2023, [arXiv:2311.09206](https://arxiv.org/abs/2311.09206).
- [23] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C.C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandez, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P.S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E.M. Smith, R. Subramanian, X.E. Tan, B. Tang, R. Taylor, A. Williams, J.X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, T. Scialom, Llama 2: Open foundation and fine-tuned chat models, 2023, [arXiv:2307.09288](https://arxiv.org/abs/2307.09288).
- [24] J. Wei, M. Bosma, V. Zhao, K. Guu, A.W. Yu, B. Lester, N. Du, A.M. Dai, Q.V. Le, Finetuned language models are zero-shot learners, in: *International Conference on Learning Representations*, 2022.
- [25] O. Hassanzadeh, N. Abdelmageed, V. Efthymiou, J. Chen, V. Cutrona, M. Hulsebos, E. Jiménez-Ruiz, A. Khatiwada, K. Korini, B. Kruit, et al., Results of SemTab 2023, in: *CEUR Workshop Proceedings*, 3557, 2023, pp. 1–14, URL: <https://ceur-ws.org/Vol-3557/paper0.pdf>.
- [26] C.S. Bhagavatula, T. Noraset, D. Downey, Tabel: Entity linking in web tables, in: *International Semantic Web Conference*, Springer, 2015, pp. 425–441.
- [27] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, Q. Liu, TinyBERT: Distilling BERT for natural language understanding, in: T. Cohn, Y. He, Y. Liu (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, Association for Computational Linguistics, Online, 2020, pp. 4163–4174.
- [28] Y. Chen, S. Qian, H. Tang, X. Lai, Z. Liu, S. Han, J. Jia, LongLoRA: Efficient fine-tuning of long-context large language models, 2024, [arXiv:2309.12307](https://arxiv.org/abs/2309.12307).
- [29] R. Avogadro, F. D'Adda, M. Cremaschi, Feature/vector entity retrieval and disambiguation techniques to create a supervised and unsupervised semantic table interpretation approach, *Knowl.-Based Syst.* 304 (2024) 112447.
- [30] R. Avogadro, F. D'Adda, M. Cremaschi, Feature/vector entity retrieval and disambiguation techniques to create a supervised and unsupervised semantic table interpretation approach, *Knowl.-Based Syst.* 304 (2024) 112447, <http://dx.doi.org/10.1016/j.knsys.2024.112447>, URL: <https://www.sciencedirect.com/science/article/pii/S0950705124010815>.
- [31] M. Paganelli, D. Tiano, F. Guerra, A multi-facet analysis of bert-based entity matching models, *VLDB J.* 33 (4) (2024) 1039–1064, <http://dx.doi.org/10.1007/s00778-023-00824-x>.
- [32] M. Naseri, P. Brandtner, R. Zimmermann, T. Falatouri, F. Darbanian, T. Obinwanne, Applications of large language models (LLMs) in business analytics—exemplary use cases in data preparation tasks, in: *International Conference on Human-Computer Interaction*, Springer, 2023, pp. 182–198.
- [33] T. Wang, X. Chen, H. Lin, X. Chen, X. Han, L. Sun, H. Wang, Z. Zeng, Match, compare, or select? An investigation of large language models for entity matching, in: *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 96–109.
- [34] J. Wang, Y. Liang, F. Meng, Z. Sun, H. Shi, Z. Li, J. Xu, J. Qu, J. Zhou, Is chatGPT a good NLG evaluator? A preliminary study, in: Y. Dong, W. Xiao, L. Wang, F. Liu, G. Carenini (Eds.), *Proceedings of the 4th New Frontiers in Summarization Workshop*, Association for Computational Linguistics, Singapore, 2023, pp. 1–11.
- [35] R. Kirk, I. Mediratta, C. Nalmpantis, J. Luketina, E. Hambro, E. Grefenstette, R. Raileanu, Understanding the effects of RLHF on LLM generalisation and diversity, in: *The Twelfth International Conference on Learning Representations*, 2024.
- [36] L. Zhong, Z. Wang, Can LLM replace stack overflow? A study on robustness and reliability of large language model code generation, *Proc. AAAI Conf. Artif. Intell.* 38 (19) (2024) 21841–21849.
- [37] Y. Zhang, M. Zhang, H. Yuan, S. Liu, Y. Shi, T. Gui, Q. Zhang, X. Huang, LLMEval: A preliminary study on how to evaluate large language models, *Proc. AAAI Conf. Artif. Intell.* 38 (17) (2024) 19615–19622.
- [38] N. Kandpal, H. Deng, A. Roberts, E. Wallace, C. Raffel, Large language models struggle to learn long-tail knowledge, in: A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, J. Scarlett (Eds.), *Proceedings of the 40th International Conference on Machine Learning*, in: *Proceedings of Machine Learning Research*, vol. 202, PMLR, 2023, pp. 15696–15707.
- [39] G. Limaye, S. Sarawagi, S. Chakrabarti, Annotating and searching web tables using entities, types and relationships, *Proc. VLDB Endow.* 3 (1–2) (2010) 1338–1347.
- [40] Z. Zhang, Effective and efficient semantic table interpretation using tableminer+, *Semant. Web* 8 (6) (2017) 921–957, <http://dx.doi.org/10.3233/SW-160242>.
- [41] D. Oliveira, M. d'Aquin, ADOG-annotating data with ontologies and graphs., in: *SemTab@ ISWC*, Vol. 2019, 2019, pp. 1–6.
- [42] B. Kruit, P. Boncz, J. Urbani, Extracting novel facts from tables for knowledge graph completion, in: *The Semantic Web – ISWC 2019*, Springer International Publishing, Cham, 2019, pp. 364–381f.
- [43] B. Steenwinckel, G. Vandewiele, F. De Turck, F. Ongenaes, Csv2kg: Transforming Tabular Data into Semantic Knowledge, *SemTab, ISWC Challenge*, 2019.
- [44] A. Thawani, M. Hu, E. Hu, H. Zafar, N.T. Divvala, A. Singh, E. Qasemi, P.A. Szekely, J. Pujara, Entity linking to knowledge graphs to infer column types and properties., in: *SemTab@ ISWC*, Vol. 2019, 2019, pp. 25–32.
- [45] M. Cremaschi, R. Avogadro, D. Chierigato, MantisTable: an automatic approach for the semantic table interpretation., in: *SemTab@ ISWC*, Vol. 2019, 2019, pp. 15–24.
- [46] M. Cremaschi, F. De Paoli, A. Rula, B. Spahiu, A fully automated approach to a complete semantic table interpretation, *Future Gener. Comput. Systems* 112 (2020) 478–500.
- [47] M. Cremaschi, R. Avogadro, A. Barazzetti, D. Chierigato, E. Jiménez-Ruiz, MantisTable SE: an efficient approach for the semantic table interpretation., in: *SemTab@ ISWC*, 2020, pp. 75–85.
- [48] V.-P. Huynh, J. Liu, Y. Chabot, T. Labbé, P. Monnin, R. Troncy, DAGOBAB: Enhanced scoring algorithms for scalable annotations of tabular data., in: *SemTab@ ISWC*, 2020, pp. 27–39.
- [49] S. Tyagi, E. Jimenez-Ruiz, Lexma: Tabular data to knowledge graph matching using lexical techniques, in: *CEUR Workshop Proceedings*, 2775, 2020, pp. 59–64, URL: <https://ceur-ws.org/Vol-2775/paper6.pdf>.
- [50] R. Shigapov, P. Zumstein, J. Kamlah, L. Oberländer, J. Mechnich, I. Schumm, Bw: Matching CSV to wikidata via meta-lookup, in: *CEUR Workshop Proceedings*, Vol. 2775, RWTH, 2020, pp. 17–26, URL: <https://ceur-ws.org/Vol-2775/paper2.pdf>.
- [51] R. Avogadro, M. Cremaschi, MantisTable v: A novel and efficient approach to semantic table interpretation., in: *SemTab@ ISWC*, 2021, pp. 79–91.
- [52] N. Abdelmageed, S. Schindler, JenTab meets SemTab 2021's new challenges., in: *SemTab@ ISWC*, 2021, pp. 42–53.
- [53] V.-P. Huynh, J. Liu, Y. Chabot, F. Deuzé, T. Labbé, P. Monnin, R. Troncy, DAGOBAB: Table and graph contexts for efficient semantic annotation of tabular data, in: *SemTab@ ISWC*, 2021, pp. 19–31.
- [54] S. Chen, A. Karaoglu, C. Negreanu, T. Ma, J.-G. Yao, J. Williams, F. Jiang, A. Gordon, C.-Y. Lin, LinkingPark: An automatic semantic table interpretation system, *J. Web Semant.* 74 (2022) 100733, <http://dx.doi.org/10.1016/j.websem.2022.100733>.

- [55] M. Cremaschi, R. Avogadro, D. Chierigato, s-elBat: a semantic interpretation approach for messy table-s, in: *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, SemTab*, Vol. 3320, CEUR-WS. org, 2022, URL: <https://ceur-ws.org/Vol-3320/paper7.pdf>.
- [56] Z. Syed, T. Finin, V. Mulwad, A. Joshi, Exploiting a web of semantic data for interpreting tables, in: *Proceedings of the Second Web Science Conference*, Vol. 5, 2010.
- [57] D. Ritze, O. Lehmberg, C. Bizer, Matching HTML tables to DBpedia, in: *5th International Conference on Web Intelligence, Mining and Semantics, WIMS '15*, ACM, New York, NY, USA, 2015, pp. 10:1–10:6.
- [58] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro, V. Christophides, Matching web tables with knowledge base entities: From entity lookups to entity embeddings, in: *The Semantic Web – ISWC 2017*, 2017, pp. 260–277.
- [59] J. Chen, E. Jiménez-Ruiz, I. Horrocks, C. Sutton, Colnet: Embedding the semantics of web tables for column type prediction, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33, 2019, pp. 29–36.
- [60] P. Nguyen, N. Kertkeidkachorn, R. Ichise, H. Takeda, Mtab: matching tabular data to knowledge graph using probability models, 2019, arXiv preprint [arXiv:1910.00246](https://arxiv.org/abs/1910.00246).
- [61] P. Nguyen, I. Yamada, N. Kertkeidkachorn, R. Ichise, H. Takeda, MTab4Wikidata at SemTab 2020: Tabular data annotation with wikidata., in: *SemTab@ ISWC*, Vol. 2775, 2020, pp. 86–95.
- [62] P. Nguyen, I. Yamada, N. Kertkeidkachorn, R. Ichise, H. Takeda, SemTab 2021: Tabular data annotation with mtab tool., in: *SemTab@ ISWC*, 2021, pp. 92–101.
- [63] H. Morikawa, Semantic table interpretation using LOD4all., in: *SemTab@ ISWC*, Vol. 2019, 2019, pp. 49–56.
- [64] R. Azzì, G. Diallo, E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, AMALGAM: making tabular dataset explicit with knowledge graph., in: *SemTab@ ISWC*, 2020, pp. 9–16.
- [65] S. Chen, A. Karaoglu, C. Negreanu, T. Ma, J.-G. Yao, J. Williams, A. Gordon, C.-Y. Lin, Linkingpark: An integrated approach for semantic table interpretation, in: *SemTab@ ISWC*, 2020.
- [66] N. Abdelmageed, S. Schindler, JenTab: Matching tabular data to knowledge graphs., in: *SemTab@ ISWC*, 2020, pp. 40–49.
- [67] W. Baazouzi, M. Kachroudi, S. Faiz, Kepler-aSI at SemTab 2021., in: *SemTab@ ISWC*, 2021, pp. 54–67.
- [68] V.-P. Huynh, Y. Chabot, T. Labbé, J. Liu, R. Troncy, From heuristics to language models: A journey through the universe of semantic table interpretation with DAGOBAB, *SemTab* (2022).
- [69] J. Liu, V.-P. Huynh, Y. Chabot, R. Troncy, Radar station: Using KG embeddings for semantic table interpretation and entity disambiguation, in: *ISWC 2022*, October 23–27, 2022, Springer, 2022, pp. 498–515.
- [70] V. Mulwad, T. Finin, Z. Syed, A. Joshi, T2LD: Interpreting and representing tables as linked data, in: *ISWC Posters & Demonstrations Track*, in: *ISWC-PD'10*, CEUR-WS.org, Aachen, Germany, 2010, pp. 25–28, URL: <https://ceur-ws.org/Vol-658/paper489.pdf>.
- [71] S. Zhang, E. Meij, K. Balog, R. Reinanda, Novel entity discovery from web tables, in: *Proceedings of the Web Conference 2020, WWW '20*, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1298–1308.
- [72] V. Mulwad, T.W. Finin, A. Joshi, Automatically generating government linked data from tables, in: *AAAI 2011*, 2011.
- [73] V. Mulwad, T. Finin, A. Joshi, Semantic message passing for generating linked data from tables, in: *The Semantic Web – ISWC 2013*, Springer Berlin Heidelberg, 2013, pp. 363–378.
- [74] C.S. Bhagavatula, T. Noraset, D. Downey, TabEL: Entity linking in web tables, in: *The Semantic Web – ISWC 2015*, 2015, pp. 425–441.
- [75] L. Yang, S. Shen, J. Ding, J. Jin, GBMTAB: A graph-based method for interpreting noisy semantic table to knowledge graph., in: *SemTab@ ISWC*, 2021, pp. 32–41.
- [76] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, X. Wu, Unifying large language models and knowledge graphs: A roadmap, *IEEE Trans. Knowl. Data Eng.* (2024) 1–20, <http://dx.doi.org/10.1109/TKDE.2024.3352100>.
- [77] Y. Li, J. Li, Y. Suhara, A. Doan, W.-C. Tan, Deep entity matching with pre-trained language models, *Proc. the VLDB Endow.* 14 (1) (2021) 50–60, <http://dx.doi.org/10.14778/3421424.3421431>.
- [78] K. Clark, M. Luong, Q.V. Le, C.D. Manning, 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.
- [79] I. Dasoulas, D. Yang, X. Duan, A. Dimou, TorchicTab: Semantic table annotation with wikidata and language models, in: *CEUR Workshop Proceedings*, 3557, 2023, pp. 21–37, URL: <https://ceur-ws.org/Vol-3557/paper2.pdf>.
- [80] K. Korini, C. Bizer, Column type annotation using ChatGPT, 2023, arXiv preprint [arXiv:2306.00745](https://arxiv.org/abs/2306.00745).
- [81] K. Korini, R. Peeters, C. Bizer, SOTAB: The WDC schema. org table annotation benchmark, in: *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, SemTab*, Vol. 3320, CEUR-WS. org, 2022, URL: <https://ceur-ws.org/Vol-3320/paper1.pdf>.
- [82] R. Peeters, C. Bizer, Using ChatGPT for entity matching, 2023, arXiv preprint [arXiv:2305.03423](https://arxiv.org/abs/2305.03423).
- [83] S. Wu, O. Irsoy, S. Lu, V. Dabrovolski, M. Dredze, S. Gehrmann, P. Kamradur, D. Rosenberg, G. Mann, BloombergGPT: A large language model for finance, 2023, [arXiv:2303.17564](https://arxiv.org/abs/2303.17564).
- [84] T. Li, S. Shetty, A. Kamath, A. Jaiswal, X. Jiang, Y. Ding, Y. Kim, CancerGPT for few shot drug pair synergy prediction using large pretrained language models, *NPJ Digit. Med.* 7 (1) (2024) 40, <http://dx.doi.org/10.1038/s41746-024-01024-9>.
- [85] Y. Sui, M. Zhou, M. Zhou, S. Han, D. Zhang, 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, WSDM '24*, Association for Computing Machinery, New York, NY, USA, 2024, pp. 645–654.
- [86] K. Kong, J. Zhang, Z. Shen, B. Srinivasan, C. Lei, C. Faloutsos, H. Rangwala, G. Karypis, OpenTab: Advancing large language models as open-domain table reasoners, in: *The Twelfth International Conference on Learning Representations*, 2024.
- [87] M.M.H. Nahid, D. Rafiei, TabSQLify: Enhancing reasoning capabilities of LLMs through table decomposition, 2024, [arXiv:2404.10150](https://arxiv.org/abs/2404.10150).
- [88] Z. Wang, H. Zhang, C.-L. Li, J.M. Eisenschlos, V. Perot, Z. Wang, L. Miculicich, Y. Fujii, J. Shang, C.-Y. Lee, T. Pfister, Chain-of-Table: Evolving tables in the reasoning chain for table understanding, in: *The Twelfth International Conference on Learning Representations*, 2024.
- [89] M. Zheng, H. Yang, W. Jiang, Z. Lin, Y. Lyu, Q. She, W. Wang, Chain-of-Thought reasoning in tabular language models, in: H. Bouamor, J. Pino, K. Bali (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, Association for Computational Linguistics, Singapore, 2023, pp. 11006–11019.
- [90] P. Li, Y. He, D. Yashar, W. Cui, S. Ge, H. Zhang, D.R. Fainman, D. Zhang, S. Chaudhuri, Table-GPT: Table-tuned GPT for diverse table tasks, 2023, [arXiv:2310.09263](https://arxiv.org/abs/2310.09263).
- [91] X. Fang, W. Xu, F.A. Tan, Z. Hu, J. Zhang, Y. Qi, S.H. Sengamedu, C. Faloutsos, Large language models (LLMs) on tabular data: Prediction, generation, and understanding - a survey, *Trans. Mach. Learn. Res.* (2024).
- [92] M. Cremaschi, F. Belotti, J. D'Souza, M. Palmonari, MammoTab 25: A large-scale dataset for semantic table interpretation – training, testing, and detecting weaknesses, in: *Proceedings of the Extended Semantic Web Conference, ESWC*, 2025, <http://dx.doi.org/10.5281/zenodo.16562700>, URL: <https://github.com/unimib-datal/mammotab/>.
- [93] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro, V. Christophides, 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*, Springer-Verlag, Berlin, Heidelberg, 2017, pp. 260–277.

- [94] N. Abdelmageed, S. Schindler, B. König-Ries, BiodivTab: A table annotation benchmark based on biodiversity research data., in: SemTab@ ISWC, 2021, pp. 13–18.
- [95] V. Cutrona, F. Bianchi, E. Jiménez-Ruiz, M. Palmonari, Tough tables: Carefully evaluating entity linking for tabular data, in: J.Z. Pan, V. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, O. Seneviratne, L. Kagal (Eds.), *The Semantic Web – ISWC 2020*, Springer International Publishing, Cham, 2020, pp. 328–343.
- [96] N. Abdelmageed, J. Chen, V. Cutrona, V. Efthymiou, O. Hassanzadeh, M. Hulsebos, E. Jiménez-Ruiz, J. Sequeda, K. Srinivas, Results of semtab 2022, in: *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching*, Vol. 3320, CEUR, 2022, URL: <https://ceur-ws.org/Vol-3320/paper0.pdf>.
- [97] R. Avogadro, M. Cremaschi, F. D'adda, F. De Paoli, M. Palmonari, LamAPI: a comprehensive tool for string-based entity retrieval with type-base filters, in: *17th ISWC Workshop on Ontology Matching, OM, 2022*, p. Online.
- [98] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, K. He, Accurate, large minibatch SGD: Training ImageNet in 1 hour, 2018, [arXiv:1706.02677](https://arxiv.org/abs/1706.02677).
- [99] A. Ibrahim, B. Thérien, K. Gupta, M.L. Richter, Q. Anthony, T. Lesort, E. Belilovsky, I. Rish, Simple and scalable strategies to continually pre-train large language models, 2024, [arXiv:2403.08763](https://arxiv.org/abs/2403.08763).
- [100] E.J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, LoRA: Low-rank adaptation of large language models, in: *International Conference on Learning Representations*, 2022.
- [101] I.J. Goodfellow, M. Mirza, D. Xiao, A. Courville, Y. Bengio, An empirical investigation of catastrophic forgetting in gradient-based neural networks, 2015, [arXiv:1312.6211](https://arxiv.org/abs/1312.6211).
- [102] M. Abidin, S.A. Jacobs, A.A. Awan, J. Aneja, A. Awadallah, H. Awadalla, N. Bach, A. Bahree, A. Bakhtiari, H. Behl, A. Benhaim, M. Bilenko, J. Bjorck, S. Bubeck, M. Cai, C.C.T. Mendes, W. Chen, V. Chaudhary, P. Chopra, A.D. Giorno, G. de Rosa, M. Dixon, R. Eldan, D. Iter, A. Garg, A. Goswami, S. Gunasekar, E. Haider, J. Hao, R.J. Hewett, J. Huynh, M. Javaheripi, X. Jin, P. Kauffmann, N. Karampatziakis, D. Kim, M. Khademi, L. Kurilenko, J.R. Lee, Y.T. Lee, Y. Li, C. Liang, W. Liu, E. Lin, Z. Lin, P. Madan, A. Mitra, H. Modi, A. Nguyen, B. Norick, B. Patra, D. Perez-Becker, T. Portet, R. Pryzant, H. Qin, M. Radmilac, C. Rosset, S. Roy, O. Ruwase, O. Saarikivi, A. Saied, A. Salim, M. Santacrose, S. Shah, N. Shang, H. Sharma, X. Song, M. Tanaka, X. Wang, R. Ward, G. Wang, P. Witte, M. Wyatt, C. Xu, J. Xu, S. Yadav, F. Yang, Z. Yang, D. Yu, C. Zhang, C. Zhang, J. Zhang, L.L. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, X. Zhou, Phi-3 technical report: A highly capable language model locally on your phone, 2024, [arXiv:2404.14219](https://arxiv.org/abs/2404.14219).
- [103] B. Pernici, C. Cappiello, C.A. Bono, C. Sancricca, T. Catarci, M. Angelini, M. Filosa, M. Palmonari, F. De Paoli, S. Bergamaschi, G. Simonini, A. Mozzillo, L. Zecchini, Sustainable quality in data preparation, *ACM J. Data Inf. Qual.* (2025) [http://dx.doi.org/10.1145/3769120](https://doi.org/10.1145/3769120).
- [104] C.A. Bono, F. Belotti, M. Palmonari, Efficient uncertainty estimation for LLM-based entity linking in tabular data, in: *Proceedings of the 20th International Workshop on Ontology Matching (OM 2025), Co-located with ISWC 2025, Nara, Japan, 2025*.

Federico Belotti is currently a Research Fellow at the University of Milano-Bicocca, specialising in machine learning techniques applied to semantic table interpretation, transformer-based models, and interpretability methods for tabular data. He earned his Master's degree in Computer Science (with honors) from the University of Milano-Bicocca, focusing his thesis on unsupervised lexical semantic change detection through compass-aligned Gaussian embeddings. Federico previously led reinforcement learning projects at Orobix S.r.l., where he developed innovative deep learning solutions and contributed significantly to open-source frameworks such as *sheeprl*. His research interests include reinforcement learning, deep learning, semantic matching, and neural interpretability. Federico has contributed to multiple peer-reviewed publications and actively participates in open-source communities.

Marco Cremaschi I am a research assistant at the University of Milan-Bicocca, Department of Informatics, Systems, and Communication (DISCo) in Italy. I hold a Ph.D. in Computer Science from the University of Milano-Bicocca (2020). During my Ph.D., I spent six months at Bielefeld University (Germany) as part of the Semantic Computing Group in 2019. Prior to that, I earned both a Bachelor's and a Master's degree in Computer Science at the University of Milano-Bicocca.

My research expertise includes knowledge graph construction, data enrichment, and, more recently, the application of artificial intelligence techniques for developing clinical decision support systems and predictive models in mental health.

Over the years, I have actively contributed to the Semantic Web and AI communities by publishing scientific papers in reputable conferences and journals. My engagement in the academic community extends to serving on the program committee of several top-tier international conferences, where I have taken on roles as chair and reviewer.

Matteo Palmonari is an Associate Professor in the Department of Informatics, Systems, and Communication at the University of Milan-Bicocca. His research spans data management and artificial intelligence, with a focus on semantic matching, knowledge graph profiling and exploration, natural language processing, and data enrichment. Recently, his interest has concentrated on the integration of symbolic and neural approaches in particular in the context of applications in the legal domain. He has played key roles in numerous innovation and research projects, serving as coordinator, scientific manager, or partner.