LLMs for Data Engineering on Enterprise Data

Jan-Micha Bodensohn*
DFKI & Technische Universität Darmstadt

Ulf Brackmann*
SAP SE & DFKI

Liane Vogel*
Technische Universität Darmstadt

Matthias Urban
Technische Universität Darmstadt

Anupam Sanghi
Technische Universität Darmstadt

Carsten Binnig
Technische Universität Darmstadt & DFKI

ABSTRACT

A recent line of work applies Large Language Models (LLMs) to data engineering tasks on tabular data, suggesting they can solve a broad spectrum of tasks with high accuracy. However, existing research primarily uses datasets based on tables from web sources such as Wikipedia, calling the applicability of LLMs for real-world enterprise data into question. In this paper, we perform a first analysis of LLMs for solving data engineering tasks on a real-world enterprise dataset. As an exemplary task, we apply recent LLMs to the task of column type annotation to study how the data characteristics affect the LLMs' accuracy and find that LLMs have severe limitations when dealing with enterprise data. Based on these findings, we point towards promising directions for adapting LLMs to the enterprise context.

VLDB Workshop Reference Format:
Jan-Micha Bodensohn, Ulf Brackmann, Liane Vogel, Matthias Urban, Anupam Sanghi, and Carsten Binnig. LLMs for Data Engineering on Enterprise Data. VLDB 2024 Workshop: Tabular Data Analysis Workshop (TaDA).

VLDB Workshop Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/DataManagementLab/llmeval-tada24.

1 INTRODUCTION

Data engineering is highly relevant. Data engineering on tabular data is crucial for transforming raw data sources into a form that is suitable for downstream tasks such as analytical query processing and machine learning. It encompasses a range of tasks spanning the entire data lifecycle, from data discovery and integration to data cleaning. As many of these tasks incur high manual efforts to apply existing tools to the specific data at hand, the automation of individual data engineering tasks such as missing value imputation [20], de-duplication [22, 23], and column type annotation [10, 12, 14, 18, 31] with the help of machine learning has long been an active area of research. Nevertheless, adapting machine learning approaches to new datasets and tasks often requires computer science expertise, thus rendering them inaccessible to a broad range of practitioners.

LLMs to the rescue? Recent papers have shown that Large Language Models (LLMs) such as GPT-4 [5] can be directly applied to data engineering tasks on tabular data, indicating that they achieve state-of-the-art results on multiple data engineering tasks without requiring task-specific architectures and training [11, 21]. Apart from these initial evaluations, further attempts have been made to adapt LLMs for specific data engineering tasks such as entity matching [24] and column type annotation [14]. Nonetheless, we do not fully share the current optimism that LLMs can solve data engineering problems on tabular data out-of-the-box because existing evaluations primarily build on tables from web sources, which do not fully represent the real-world complexity of tabular data.

Enterprise data: An overlooked challenge. Tables in publicly available corpora are often crawled from web resources like Wikipedia [2] and GitHub [9]. However, tabular data from companies running their business processes with software systems like those from SAP fundamentally differs from these web tables in many aspects, including table sizes, data types, and industry-specific domains [25, 30]. Since LLMs are typically trained on public data scraped from the web [1, 3], it is reasonable to assume that they have not seen significant amounts of such enterprise data during their training, which may cause limited understanding. We, therefore, suspect that previous evaluation results on publicly available datasets do not extend to real-world enterprise settings.

Contributions. In this work, we thus perform a first evaluation of LLMs for data engineering on real-world enterprise data. The concrete contributions of this paper are: (1) We analyze the characteristics of enterprise data from real-world customer systems in comparison to publicly available table corpora. (2) We perform experiments on this data using the task of column type annotation as a first example to expose the shortcomings of LLMs on enterprise data. (3) We provide directions for future research to improve the automation of data engineering tasks on enterprise data.

2 ANATOMY OF ENTERPRISE DATA

In this section, we make an attempt to quantify the anatomy of enterprise data using real-world customer data from SAP systems. While there are clearly many more enterprise systems besides SAP, SAP stands out as a dominating player in enterprise software systems across multiple industries worldwide. Moreover, enterprise

*Authors with equal contribution, alphabetical ordering. Correspondence goes to jan-micha.bodensohn@cs.tu-darmstadt.de.

Proceedings of the VLDB Endowment. ISSN 2150-8097.

Table 1: Data characteristics of publicly available web table corpora compared to representative customer data from SAP. Enterprise tables are substantially larger in terms of rows and columns and display a higher sparsity. Although most attributes are of type NVARCHAR, the data is highly symbolic.

	#tables	#columns		#rows		sparsity[1]	data types[2]		column type annotations	
		med	95th	med	95th		*abc*	*123*	#column types	#labeled columns
WikiTables-TURL	397, 098	1	3	8	43	0.12	1.00	0.00	255	628, 254
SOTAB	59, 548	7	17	33	721	0.08	0.85	0.15	91	162, 351
GitTablesCTA	1, 100	12	33	25	263	0.12	0.33	0.67	122 | 59[3]	2, 517 | 1, 374[3]
SportsTables	1, 183	21	31	32	924	0.07	0.16	0.84	452	24, 821
EnterpriseTables	100[4]	46	341	384, 320	43, 758, 473	0.43	0.86[5]	0.14	5, 063	8, 066

[1] Sparsity is the fraction of empty table cells. [2] Non-numeric *(abc)* and numeric *(123)* columns determined by pandas (publicly available datasets) and SQL *(EnterpriseTables).*
[3] Annotations using semantic types from DBPedia | Schema.org. [4] We experiment on a representative sample from multiple thousand tables from the real-world system.
[5] Includes symbolic values such as IDs stored as NVARCHAR.

data is usually highly confidential and, therefore, hard to use in evaluations. As such, we believe that our insights based on SAP data are highly valuable on their own and hope that our paper inspires other researchers with access to similar enterprise datasets to repeat the evaluations on their data. Moreover, some of the core characteristics of SAP data reflect the findings from other papers observing the differences between enterprise and web data [13, 25, 30, 32].

Our corpus. To study the differences between web tables and enterprise data, we create a new corpus called *EnterpriseTables* using the real-world customer data from SAP. The corpus spans a diverse set of business domains such as Finance, Sales and Distribution, Material Management, and Production Planning. For the purpose of this paper, we select 100 representative tables from the larger corpus, which contains multiple thousands of tables. In the following, we compare this new corpus to four existing publicly available table corpora, namely WikiTables-TURL [4], SOTAB [15], GitTablesCTA [8], and SportsTables [17].

Tables are substantially larger and wider. A first important observation is that enterprise tables typically have substantially more rows and columns than the tables in public corpora. As shown in Table 1, some tables have hundreds of columns and millions of rows. While this is a well-established data management problem [32], the large scale poses challenges for LLMs, which have limited context windows. Moreover, while recent LLMs have extended context windows, feeding large tables into LLMs has other downsides since not only latency and cost depend on the input size, but also recent studies have shown that long contexts can lead to degraded performance for data residing in the "middle" [19].

Tables are highly sparse. A second insight is that enterprise data is highly sparse. Table 1 shows that on average, 43% of the cells in enterprise tables are empty, compared to only 7-12% in existing datasets. Moreover, we find that in addition to empty values, the cells in enterprise tables often contain dummy values such as 00000 that also denote the absence of an actual value.

Schemas are not descriptive. Another important insight replicated here is that schema properties like table and column names are often not descriptive but rather abbreviations that can only be understood with background knowledge or additional metadata [13]. This additional metadata is often unavailable or may not fit into the context window of the LLM. Moreover, the background knowledge is often specific to the particular enterprise, causing challenges for LLMs trained exclusively on publicly available data.

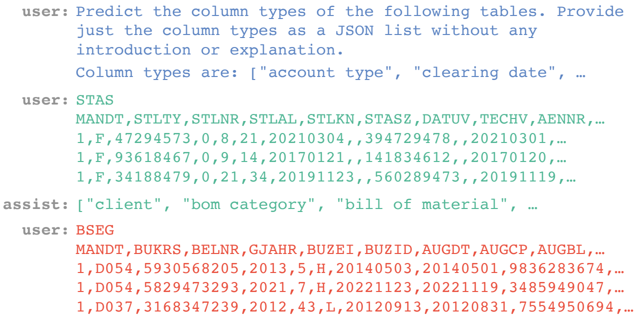
Data are complex. Surprisingly, we find that only 14% of the columns are of numerical data types such as DECIMAL and INTEGER, challenging the popular assumption that enterprise data is predominantly numerical [17]. Nevertheless, closer inspection of the actual data reveals that the non-numerical data type NVARCHAR is often used to store symbolic values and codes such as invoice and material numbers, which is in line with previous findings [30]. Therefore, we believe that a more fine-grained investigation of values beyond numeric and non-numeric data types is needed in future work.

Entities are represented by multiple tables. Finally, a last important characteristic is that whereas existing tabular datasets are usually collections of self-contained tables, data in enterprise contexts typically describes business objects such as invoices and orders that span across multiple connected tables. For example, in SAP systems, data pertaining to a particular material is scattered across the MARA (material type and basic statistics), MARC (manufacturing-related details), MBEW (valuation data), and other tables. As a result, many data engineering tasks such as entity matching cannot be solved based on individual tables since the complete relational context and structure must be considered.

3 INITIAL EVALUATION

In this section, we empirically examine the challenges that arise when applying LLMs to data engineering on enterprise data.

Task. For our initial evaluation, we focus on column type annotation (CTA), a well-established task whose goal is to annotate the columns of a relational table with semantic types from a pre-defined ontology such as DBPedia [10, 12, 14, 18, 31]. We choose CTA as a first exemplary task for our analysis since it requires a semantic understanding of the content of every column as well as its tabular context, such as the values of other columns [27, 31]. This fine granularity of the CTA task allows us to demonstrate many of the challenges that arise on enterprise data.



```
  user: Predict the column types of the following tables. Provide
        just the column types as a JSON list without any
        introduction or explanation.
        Column types are: ["account type", "clearing date", …
  user: STAS
        MANDT,STLTY,STLNR,STLAL,STLKN,STASZ,DATUV,TECHV,AENNR,…
        1,F,47294573,0,8,21,20210304,,394729478,,20210301,…
        1,F,93618467,0,9,14,20170121,,141834612,,20170120,…
        1,F,34188479,0,21,34,20191123,,560289473,,20191119,…
assist: ["client", "bom category", "bill of material", …
  user: BSEG
        MANDT,BUKRS,BELNR,GJAHR,BUZEI,BUZID,AUGDT,AUGCP,AUGBL,…
        1,D054,5930568205,2013,5,H,20140503,20140501,9836283674,…
        1,D054,5829473293,2021,7,H,20221123,20221119,3485949047,…
        1,D037,3168347239,2012,43,L,20120913,20120831,7554950694,…
```

Figure 1: Example prompt for CTA with instruction, one-shot example, and table to annotate. The data are fictional.

Datasets. We experiment on two existing CTA datasets as well as our novel *EnterpriseTables* corpus:

1) *GitTablesCTA* [8] is a subset of the GitTables corpus [9] annotated with semantic types. For our experiments, we use 350 tables annotated with 122 semantic types from DBPedia.
2) *SportsTables* [17] consists of web tables describing various sport events. We include this dataset for its high proportion of numeric data and large number of semantic types. We use 500 randomly selected tables annotated with 452 semantic types.
3) *EnterpriseTables* has 100 tables annotated with 5, 063 semantic types derived from the SAP data dictionary, like `client` and `material category`. To make the large number of semantic types tractable, we include only a subset of the semantic types in every prompt.

Models. We use GPT-3.5-Turbo and GPT-4[1] from OpenAI [5]. The GPT models are known for their high performance, and comparing GPT-3.5-Turbo to GPT-4 demonstrates the benefits of the different generations. GPT-3.5-Turbo has a context window of 16, 385 tokens and a cost of 0.5$ per 1M input tokens, whereas GPT-4 has a context window of 8, 192 tokens and a cost of 30$ per 1M input tokens. In the future, we plan to apply additional models, including recent open source models.

Prompting. Our prompting strategy builds on best practices from existing literature, which we had to adapt to the unique characteristics of the enterprise tables. Figure 1 shows an example prompt consisting of a short instruction, a list of all semantic types, one randomly selected example, and the table to annotate. In our evaluation, we use a task formulation where the model annotates all columns of the given table, which is a setting chosen also by other papers [14, 31]. However, due to the LLMs' limited context windows, we have to limit each table to three randomly selected rows. For similar reasons, we serialize the tables in CSV format, which requires fewer formatting tokens than other serialization schemes like Markdown and JSON [26, 28]. Finally, we instruct the model to generate the column types as a JSON-formatted list.

Table 2: Enterprise vs. web tables. The table shows support-weighted F1 scores for CTA with and without column names. The results on enterprise data are substantially worse than on existing benchmarks.

	EnterpriseTabs		GitTablesCTA		SportsTables	
column names	**w/out**	**with**	**w/out**	**with**	**w/out**	**with**
GPT-3.5-Turbo	0.02	0.08	0.39	0.82	0.32	0.62
GPT-4	0.03	0.17	0.55	0.98	0.58	0.90

Table 3: Non-numeric *(abc)* vs. numeric *(123)* data. The table shows support-weighted F1 scores for CTA with column names. Results on numeric data are on par with or worse than on non-numeric data across all models and datasets.

	EnterpriseTabs		GitTablesCTA		SportsTables	
data types	***abc***	***123***	***abc***	***123***	***abc***	***123***
GPT-3.5-Turbo	0.09	0.05	0.84	0.80	0.76	0.60
GPT-4	0.18	0.15	0.98	0.98	0.91	0.90

3.1 Exp. 1: Overall Performance

In our first experiment, we compare the performance for CTA on our new *EnterpriseTables* corpus with the performance on GitTablesCTA and SportsTables. We perform each experiment twice, with and without including the table and column names (i.e., the table schema). While existing evaluations typically leave out the column names since the semantic types are directly derived from them and the task would thus become trivial, for *EnterpriseTables*, the CTA task is much harder. Therefore, we want to investigate how much the additional information helps.

Results. Table 2 shows the results of this experiment. We make the following key observations: (1) LLMs have severe problems with CTA on enterprise data, leading to substantially worse results compared to the web resources GitTablesCTA and SportsTables. Especially in the experiments without column names, the results on enterprise data are particularly poor (F1 scores of only 0.02 and 0.03), indicating that the enterprise data on its own contains few helpful signals. (2) Adding column names to the prompt improves the results on the enterprise data, but only up to 0.17 using GPT-4, which is still much lower than for web tables. The remaining performance gap could potentially be attributed to the cryptic schema, extremely wide and sparse tables, and the complex data types described in Section 2. (3) Finally, we observe that GPT-4 performs substantially better than GPT-3.5-Turbo across all datasets but still cannot handle enterprise data well.

3.2 Exp. 2: Impact of Numerical Data

LLMs are known to often perform better on textual data than on numerical data [6, 17]. To analyze this effect on enterprise data, we examine the performance for numeric and non-numeric columns in our *EnterpriseTables* corpus.

[1] We use `gpt-3.5-turbo-1106` and `gpt-4-0613`.

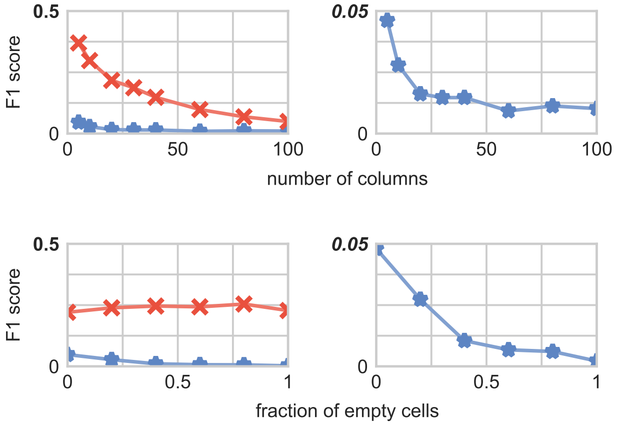


Figure 2: Varying numbers of columns (top) and sparsities (bottom). The plots show support-weighted F1 scores for CTA with GPT-3.5-Turbo on *EnterpriseTables* prompted with × and without ∗ column names (zoomed in on the right). Increasing numbers of columns lead to worse results. Increased sparsity leads to worse results if no column names are given.

Results. Table 3 shows that, as expected, we see a higher performance on non-numeric data. By contrast, using GPT-4 on the GitTablesCTA and SportsTables datasets, we do not observe performance differences between the data types. The low performance on the *EnterpriseTables* dataset even with GPT-4 (F1 scores of 0.18 for non-numeric and 0.15 for numeric data) indicates that numeric data in enterprise systems is even harder to understand than in web tables. Furthermore, the low accuracy for non-numeric columns could stem from the fact that the enterprise tables often store identifiers like `0014056` as type `NVARCHAR`, as we explained in Section 2.

3.3 Exp. 3: Table Width and Sparsity

To investigate the performance gap between LLMs applied to web tables compared to enterprise data, we incrementally adapt the enterprise tables to resemble the characteristics of web tables more closely. As two of the main differences are table width and sparsity, we vary the number of columns per table by randomly sampling subsets of columns (Figure 2 top), and vary the sparsity by incrementally sparsifying the columns (Figure 2 bottom).

Results. As we can see in Figure 2, increasing numbers of columns lead to substantially worse results, indicating that the large table widths in enterprise data are indeed a major problem for LLMs. While one solution is splitting up large tables into multiple smaller column sets, important context information may then be lost. Regarding sparsity, we observe worse results with increased sparsity if no column names are provided, whereas with column names, increased sparsity does not change the results much. This indicates that LLMs primarily rely on the column headers to predict the semantic types and do not take the cell values into account.

4 THE ROAD AHEAD

With this paper, we make a first attempt to illustrate the challenges of applying LLMs to real-world enterprise data. As shown in our initial study, current state-of-the-art LLMs do not work out-of-the-box on enterprise data as they do on web tables due to the unique characteristics of the enterprise data. In the following, we discuss potential directions to close this gap.

Improving LLMs for CTA on enterprise data. To improve the performance of LLMs when predicting semantic column types on enterprise data, future work could look into prompt engineering [14] and fine-tuning [7] with the characteristics of the enterprise data in mind. Moreover, helpful context information included in the prompt, such as example values for each column type, could further support the model. Finally, models such as Pythagoras [16] that are specifically designed to handle, for example, numeric data, are a promising research direction.

Tackling other data engineering tasks. Apart from column type annotation, there are many more data engineering tasks to automate [11, 21]. Many of these tasks come with additional challenges on real-world enterprise data. For example, entity matching is defined in the literature as identifying the rows in two tables that refer to the same real-world entity [24]. However, in the context of enterprise data, the data describing a single entity (business object) is usually scattered across multiple tables. Hence, to decide whether two business objects match, a model must incorporate the values stored in multiple connected tables instead of just one table. This makes entity matching and other tasks like error detection and missing value imputation tricky to address using LLMs since such relationships across multiple tables are hard to capture using natural language prompts and limited context windows.

Synthetic enterprise data. As mentioned, enterprise data is typically highly confidential and, thus, cannot be made available to the public. One way to overcome this issue could be to adapt existing table corpora to more closely resemble the characteristics of enterprise data laid out in Section 2, for example by obfuscating column names or by extending tables with additional rows and columns.

The need for Tabular Foundation Models. Finally, we advocate the development of new tabular foundation models designed with the characteristics of enterprise data in mind. For example, to address the complex structure of enterprise data, a promising way forward is to combine LLMs with graph neural networks that can take the complete relational structure into account [29]. Such models could incorporate the values stored in different tables to decide whether two entities refer to the same real-world concept.

ACKNOWLEDGMENTS

This work has been supported by the BMBF and the state of Hesse as part of the NHR Program and the HMWK cluster project 3AI. It was also partially funded by the LOEWE Spitzenprofessur of the state of Hesse. We also thank DFKI Darmstadt and hessian.AI as well as Isil Pekel, Dr. Sam Thelin, and Roberto Falk from SAP for their support.

REFERENCES

[1] AI@Meta. 2024. Llama 3 Model Card. (2024). https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[2] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2015. TabEL: Entity Linking in Web Tables. In *The Semantic Web - ISWC 2015*. Springer International Publishing, Cham, 425–441. https://doi.org/10.1007/978-3-319-25007-6_25

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[4] Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. TURL: table understanding through representation learning. *Proceedings of the VLDB Endowment* 14, 3 (Nov. 2020), 307–319. https://doi.org/10.14778/3430915.3430921

[5] OpenAI et al. 2024. GPT-4 Technical Report. http://arxiv.org/abs/2303.08774 arXiv:2303.08774 [cs].

[6] Simon Frieder, Luca Pinchetti, Alexis Chevalier, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Petersen, and Julius Berner. 2023. Mathematical Capabilities of ChatGPT. 37th Conference on Neural Information Processing Systems (NeurIPS 2023) Track on Datasets and Benchmarks (2023).

[7] Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. (2022).

[8] Madelon Hulsebos, Çağatay Demiralp, and Paul Demiralp. 2021. GitTables benchmark - column type detection. https://doi.org/10.5281/zenodo.5706316

[9] Madelon Hulsebos, Cagatay Demiralp, and Paul Groth. 2023. GitTables: A Large-Scale Corpus of Relational Tables. *Proceedings of the ACM on Management of Data* 1, 1 (May 2023), 30:1–30:17. https://doi.org/10.1145/3588710

[10] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Cagatay Demiralp, and César Hidalgo. 2019. Sherlock: A Deep Learning Approach to Semantic Data Type Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. Association for Computing Machinery, New York, NY, USA, 1500–1508. https://doi.org/10.1145/3292500.3330993

[11] Gonzalo Jaimovitch-López, Cèsar Ferri, José Hernández-Orallo, Fernando Martínez-Plumed, and María José Ramírez-Quintana. 2022. Can language models automate data wrangling? *Machine Learning* 112, 6 (2022), 2053–2082. https://doi.org/10.1007/s10994-022-06259-9

[12] Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, and Kavitha Srinivas. 2020. SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. In *The Semantic Web*. Springer International Publishing, Cham, 514–530. https://doi.org/10.1007/978-3-030-49461-2_30

[13] Jaewoo Kang and Jeffrey F. Naughton. 2003. On schema matching with opaque column names and data values. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data (SIGMOD '03)*. Association for Computing Machinery, New York, NY, USA, 205–216. https://doi.org/10.1145/872757.872783

[14] Keti Korini and Christian Bizer. 2023. Column Type Annotation using ChatGPT. In *Joint Proceedings of Workshops at the 49th International Conference on Very Large Data Bases (VLDB 2023), Vancouver, Canada, August 28 - September 1, 2023 (CEUR Workshop Proceedings)*, Vol. 3462. CEUR-WS.org. https://ceur-ws.org/Vol-3462/TADA1.pdf

[15] Keti Korini, Ralph Peeters, and Christian Bizer. 2022. SOTAB: The WDC Schema.org Table Annotation Benchmark. *SemTab@ISWC 2022, October 23–27, 2022, Hangzhou, China (Virtual)* 3320 (2022), 14–19.

[16] Sven Langenecker, Christoph Sturm, Christian Schalles, and Carsten Binnig. 2023. Pythagoras: Semantic Type Detection of Numerical Data Using Graph Neural Networks. *Proceedings of the 27th International Conference on Extending Database Technology (EDBT), 25th March-28th March, 2024*, (2023).

[17] Sven Langenecker, Christoph Sturm, Christian Schalles, and Carsten Binnig. 2023. SportsTables: A new Corpus for Semantic Type Detection. (2023). https://doi.org/10.18420/BTW2023-68 ISBN: 9783885797258 Publisher: Gesellschaft für Informatik e.V.

[18] Sven Langenecker, Christoph Sturm, Christian Schalles Schalles, and Carsten Binnig. 2023. Steered Training Data Generation for Learned Semantic Type Detection. *Proceedings of the ACM on Management of Data* 1, 2 (June 2023), 1–25. https://doi.org/10.1145/3589786

[19] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Trans. Assoc. Comput. Linguistics* 12 (2024), 157–173. https://doi.org/10.1162/TACL_A_00638

[20] Yinan Mei, Shaoxu Song, Chenguang Fang, Haifeng Yang, Jingyun Fang, and Jiang Long. 2021. Capturing Semantics for Imputation with Pre-trained Language Models. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, Chania, Greece, 61–72. https://doi.org/10.1109/ICDE51399.2021.00013

[21] Avanika Narayan, Ines Chami, Laurel Orr, and Christopher Ré. 2022. Can Foundation Models Wrangle Your Data? *Proceedings of the VLDB Endowment* 16, 4 (Dec. 2022), 738–746. https://doi.org/10.14778/3574245.3574258

[22] Felix Naumann and Melanie Herschel. 2010. *An Introduction to Duplicate Detection*. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-031-01835-0

[23] Thorsten Papenbrock, Arvid Heise, and Felix Naumann. 2015. Progressive Duplicate Detection. *IEEE Transactions on Knowledge and Data Engineering* 27, 5 (May 2015), 1316–1329. https://doi.org/10.1109/TKDE.2014.2359666

[24] Ralph Peeters and Christian Bizer. 2023. Using ChatGPT for Entity Matching. In *New Trends in Database and Information Systems - ADBIS 2023 Short Papers, Doctoral Consortium and Workshops: AIDMA, DOING, K-Gals, MADEISD, PeRS, Barcelona, Spain, September 4-7, 2023, Proceedings (Communications in Computer and Information Science)*, Vol. 1850. Springer, 221–230. https://doi.org/10.1007/978-3-031-42941-5_20

[25] Alexandra Savelieva, Andreas Mueller, Avrilia Floratou, Carlo Curino, Hiren Patel, Jordan Henkel, Joyce Cahoon, Markus Weimer, Nellie Gustafsson, Richard Wydrowski, Roman Batoukov, Shaleen Deep, and Venkatesh Emani. 2022. The Need for Tabular Representation Learning: An Industry Perspective. *Table Representation Learning Workshop at NeurIPS 2022* (2022).

[26] Ananya Singha, José Cambronero, and Sumit Gulwani. 2023. Tabular Representation, Noisy Operators, and Impacts on Table Structure Understanding Tasks in LLMs. *Table Representation Learning Workshop at NeurIPS 2023* (2023).

[27] Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. 2022. Annotating Columns with Pre-trained Language Models. In *Proceedings of the 2022 International Conference on Management of Data*. ACM, Philadelphia PA USA, 1493–1503. https://doi.org/10.1145/3514221.3517906

[28] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024. 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*. ACM, Merida Mexico, 645–654. https://doi.org/10.1145/3616855.3635752

[29] Liane Vogel, Benjamin Hilprecht, and Carsten Binnig. 2022. Towards Foundation Models for Relational Databases [Vision Paper]. *Table Representation Learning Workshop at NeurIPS 2022* (2022), 6.

[30] Adrian Vogelsgesang, Michael Haubenschild, Jan Finis, Alfons Kemper, Viktor Leis, Tobias Muehlbauer, Thomas Neumann, and Manuel Then. 2018. Get Real: How Benchmarks Fail to Represent the Real World. In *Proceedings of the Workshop on Testing Database Systems*. ACM, Houston TX USA, 1–6. https://doi.org/10.1145/3209950.3209952

[31] Dan Zhang, Madelon Hulsebos, Yoshihiko Suhara, Çağatay Demiralp, Jinfeng Li, and Wang-Chiew Tan. 2020. Sato: contextual semantic type detection in tables. *Proceedings of the VLDB Endowment* 13, 12 (Aug. 2020), 1835–1848. https://doi.org/10.14778/3407790.3407793

[32] Erkang Zhu, Dong Deng, Fatemeh Nargesian, and Renée J. Miller. 2019. JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes. In *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 847–864. https://doi.org/10.1145/3299869.3300065

Received 23 May 2024; accepted 28 June 2024