TabHD: Table Healer Dataset for OCR-Damaged Table Restoration with LLMs

Anonymous ACL submission

Abstract

Tabular data contains structural information in the form of rows and columns and is utilized across various industries, including finance, government, and science. However, during the Optical Character Recognition (OCR) process, table structures can become distorted, or cell values may be extracted incorrectly, leading to a decline in the performance of subsequent tasks that rely on tabular data. Existing OCR post-processing techniques primarily focus on general text recovery, which limits their effectiveness in restoring complex table structures. To address this issue, we have constructed a large-scale benchmark dataset called TabHD for recovering tabular data damaged after OCR processing. Using this dataset, we systematically evaluate the table restoration performance of Large Language Models (LLMs). This study explores the potential of LLM-based table restoration in the OCR post-processing pipeline and suggests directions for the development of more sophisticated models in the future.

1 Introduction

Tabular data consists of rows and columns, incorporating both structural information and semantic information based on cell content. It is widely utilized across various industries, including finance, government, and science, for efficient data storage and analysis (Shigarov, 2023). With recent advancements in deep learning technologies, there has been increasing research into effectively processing tabular data.

Optical Character Recognition (OCR) is the process of converting images into machine-readable text and includes document layout analysis (DLA) and table data extraction as its primary tasks. Through OCR, not only continuous text in images but also tabular data can be extracted as text (Patel et al., 2012). However, table structures may not always be extracted in their original form (Peng et al., 2024). For instance, long tables may be split into multiple CSV files, or multiple adjacent tables may be mistakenly recognized as a single table. Additionally, errors frequently occur in cells containing a mix of numerical and natural language data, leading to incorrect value extraction (Patel et al., 2012). Such errors can degrade the performance of downstream tasks that rely on tabular data. While some post-OCR techniques exist to correct spelling errors, there is still a lack of models dedicated to restoring table structures.

With the recent advancement of Large Language Models (LLMs), various text-based tasks can now be performed. LLMs have been utilized for handling tabular data in formats such as HTML, JSON, and CSV, and models like TableLlama and TableGPT have been introduced for table-based tasks, including Table Interpretation, Table Augmentation, and Question Answering (Zhang et al., 2023). However, LLMs still struggle to fully comprehend and process structured tabular data. In particular, research on Table Recovery, a complex data restoration task, remains insufficient.

To address post-OCR challenges, we propose TabHD, a large-scale benchmark dataset for tabular data recovery. TabHD is constructed based on existing open-source tabular datasets, including the table-to-text (ToTTo) dataset (Parikh et al., 2020), GitTables (Hulsebos et al., 2023), and the BioTable dataset from the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab) 2021 (Abdelmageed et al., 2021). It is designed to reflect various error scenarios while maintaining the consistency of table structures and cell content.

Furthermore, we experimentally validate an approach where table structural information is converted into a graph representation and provided as additional input to LLMs. To achieve this, we fine-tuned LLMs on the TabHD dataset through instruction tuning, enabling the model to restore misrecognized table structures and correct incor-

rectly extracted cell values.

The key contributions of this study are as follows: 1. TabHD: We introduce the first large-scale benchmark dataset for OCR post-processing. TabHD consists of 42,033 tables encompassing diverse error scenarios.

2. Systematic evaluation of existing LLMs for table restoration: We analyze the impact of incorporating table structural information on model performance.

2 Related Works

2.1 LLM for Tabular Data

Tables represent structured data, requiring consideration of both textual meaning and positional information, making table-related tasks highly challenging. With the recent advancements in Large Language Models (LLMs), research on enabling LLMs to perform tasks based on tabular data has been actively conducted.

Sui et al. designed a benchmark to evaluate the Structured Understanding Capability (SUC) of LLMs, analyzing the performance of GPT-3.5 and GPT-4 across seven tasks, including cell lookup, row retrieval, and size detection. They reported performance improvements in various table-based tasks by optimizing structured prompts using a self-augmentation technique (Sui et al., 2024a).

TableLlama, an LLM built on Llama 2 7B and LongLoRA, is designed to handle table-based tasks such as Table Interpretation, Table Augmentation, and Question Answering (Zhang et al., 2023). Similarly, TableGPT fine-tunes LLMs with external functional commands, enabling table-related tasks such as Table Question Answering, data manipulation (e.g., insertion and deletion), and data visualization (Zha et al., 2023).

Despite these advancements, LLMs still lack the ability to fully comprehend and process structured tabular data (Sui et al., 2024b).

2.2 Graph LLM Fusion

Graph networks are effective for representing structured data, as they can capture both the values of individual table cells and their relationships with surrounding cells. Recently, there has been a growing body of research integrating graph structures with language models.

GraphPrompter is a novel framework that combines graph neural networks (GNNs) with soft prompts to effectively convey graph information to LLMs, thereby improving their predictive capabilities in graph-related tasks (Liu et al., 2024). Carta et al. propose an innovative approach that leverages generative LLMs, such as GPT-3.5, to address key challenges in knowledge graph construction. Their method enables scalable and flexible automated knowledge graph generation by employing an iterative zero-shot prompting strategy while reducing dependency on external knowledge (Carta et al., 2023).

Graph-ToolFormer explores a learning method in which LLMs leverage external graph reasoning tools to enhance logical inference over graph data (Zhang, 2023). ProLINK is a framework that enhances pre-trained GNNs by utilizing graph-structured prompts generated by LLMs, significantly improving low-resource inductive reasoning performance across various knowledge graphs (KGs) without requiring additional training (Wang et al., 2024a). KGP (Knowledge Graph Prompting), on the other hand, proposes a framework that enhances LLM performance in multi-document question answering (MD-QA) by utilizing KG-based prompts to effectively structure and explore logical relationships between documents (Wang et al., 2024b).

While these studies share the common goal of integrating LLMs with graph information to enhance structured data processing and reasoning capabilities, they differ in their approach. GraphPrompter and ProLINK focus on graph-based reasoning using GNNs, Carta et al.'s method and KGP emphasize knowledge graph generation and utilization via LLMs, and Graph-ToolFormer highlights the collaboration between LLMs and external graph reasoning tools(Liu et al., 2024; Carta et al., 2023; Zhang, 2023; Wang et al., 2024b).

3 TabHD Benchmark

3.1 Dataset Collection

To create the TabHD benchmark, we collected a total of three open-source tabular datasets. The first dataset is the table-to-text (ToTTo) dataset, which contains over 120,000 English tables (Parikh et al., 2020). It was originally collected for the purpose of generating descriptive text for tabular data, where each table is paired with corresponding text. However, in this study, we used only the tabular data. The second dataset is GitTables, an open-source dataset that consists of approximately one million CSV tables collected from GitHub (Hulsebos et al.,

Dataset	Table Count
ToTTo	128461
GitTables	350584
BioTable	110

Table 1: Collected Table Dataset Count

2023). Finally, we utilized the BioTable dataset from the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab) 2021 (Abdelmageed et al., 2021). The number of tables in each dataset is shown in Table 1, and a subset of these datasets was used to construct TabHD.

3.2 Task Formulation and TabHD Construction

Most existing datasets are designed for question-answering tasks based on tabular data. However, TabHD was created with a different objective: to determine whether LLMs can restore damaged tabular data in text format. To achieve this, we defined four tasks: 1) Table Merge, 2) Table Split, 3) Table Cell Imputation, and 4) Table Cell Correction. The collected datasets were preprocessed to align with these four defined tasks, forming the TabHD benchmark. The number of datasets used for each task is shown in Table 2, with approximately 50,000 tables allocated per task.

3.2.1 Table Merge

Most tabular data appears in documents alongside various layouts, including text and images. Due to this, length limitations may cause table data to be truncated. When a table is split across different pages, OCR tends to recognize each fragment as an independent table. To utilize the original, undamaged tabular data, it is necessary to restore these fragmented tables to their original structure. If only a single table exists in a document, simply merging the fragments is sufficient. However, if multiple fragmented tables exist within the same document, a criterion is needed to determine which table fragments should be merged. Therefore, we define the Table Merge task to recover tables that have been divided into multiple fragments.

To implement the Table Merge task, we used a total of 61,645 tables. To simulate fragmented table data, we split 30,844 tables into two parts and 30,801 tables into three parts. When splitting the tables, half of them were divided randomly based on rows, while the other half were divided based on columns.

For the tables that were split into two parts, we created three different input settings for the language model. In 5,144 cases, only one table fragment was provided as input. In 10,280 cases, both fragments of a split table were included as input. In 15,420 cases, three separate table fragments were provided simultaneously as input. This setup was designed to evaluate the model's ability to determine which table fragments should be correctly merged when multiple fragmented tables exist in a single document.

Similarly, for the tables split into three parts, 5,136 cases contained a single table fragment as input, 10,266 cases included two fragments, and 15,399 cases provided all three fragments simultaneously as input to the language model.

3.2.2 Table Split

Another issue that arises when extracting tabular data from documents using OCR is that multiple adjacent tables may be mistakenly recognized as a single table. If several tables are placed closely together, OCR often merges them into one. For example, if three tables are arranged on a single page for numerical comparison, OCR may interpret them as a single table. To address this issue, we define the Table Split task, which aims to determine whether merged tables can be correctly separated into their original independent forms.

To implement the Table Split task, we used a total of 51,839 tables. To simulate cases where multiple tables are incorrectly recognized as one, we merged 25,851 tables into pairs, combining them into a single table. Half of these were merged vertically, while the other half were merged horizontally. Additionally, 15,988 tables were grouped into sets of three and merged into a single table.

3.2.3 Table Cell Imputation

There are various factors that contribute to the degradation of OCR performance, including font style and image quality. If the OCR model encounters an unfamiliar font or if the text is difficult to recognize, missing values may occur in the table's cells. Tabular data consists of multiple cells, each containing not only its own content but also structural information related to its rows and columns. Therefore, when missing values occur in table cells, it can lead to issues in downstream tasks such as numerical prediction and categorical classification. To address this, we define the Table Cell Imputation task, which aims to determine whether missing

Task	Sub-Task Category	Sub-Task	Table Count
Table Merging	Merging a table split into two	1 table fragment	5144
		2 table fragments	10280
		3 table fragments	15420
	Merging a table split into three	1 table fragment	5136
		2 table fragments	10266
		3 table fragments	15399
Table Splitting	Splitting a merged table	Splitting into two tables	25851
		Splitting into three tables	25988
Table Cell Completion	Filling missing values	10% missing	18248
		30% missing	18239
		50% missing	18249
Table Cell Correction	Fixing ambiguous characters	10% affected	5245
		30% affected	5300
		50% affected	5291
	Restoring truncated text	10% affected	5173
		30% affected	5010
		50% affected	5090
	Restoring reordered text	10% affected	5237
		30% affected	5120
		50% affected	5132
Total			**214818**

Table 2: Table task distribution with sub-task categories and table counts.

values in tabular data can be effectively restored.

To implement the Table Cell Imputation task, we utilized a total of 54,736 tables. Based on the proportion of missing values per table, 18,248 tables were assigned a 10 percent missing rate, 18,239 tables were assigned a 30 percent missing rate, and 18,249 tables were assigned a 50 percent missing rate.

3.2.4 Table Cell Correction

Despite significant advancements in OCR technology, certain characters are still prone to recognition errors. For example, OCR often misinterprets 'B' as '8' or '0' as 'O', leading to recognition inaccuracies for visually similar characters (reference needed). Additionally, as the length of text increases, OCR performance tends to decline (reference needed). A common issue with long text strings is that OCR may alter the order of middle characters or omit certain characters. However, when tabular data is used after OCR processing, damaged cell content can lead to performance degradation in downstream tasks. To address this issue, we define the Table Cell Correction task, which aims to determine whether corrupted cell content can be accurately restored.

For the Table Cell Correction task, we used a total of 46,598 tables. Among these, 15,836 tables were modified by replacing characters that OCR frequently misrecognizes. The specific characters with low OCR recognition accuracy used in this task are shown in Table 2 (table reference needed).

For 15,273 tables, we simulated scenarios where OCR truncates longer text by cutting off text or numerical values when they exceeded four characters in length.

For 15,489 tables, we implemented cases where the character order was altered, reflecting common OCR errors in handling long text strings.

For all three scenarios, the number of tables was evenly divided into three groups, with 10 percent, 30 percent, and 50 percent of the cell content intentionally corrupted.

4 Experiments

Tables extracted through OCR often fail to preserve their original structure, leading to various issues in subsequent data utilization processes. To address these challenges, this study proposes a table restoration model that integrates LLMs and GNNs and aims to validate its effectiveness through experiments.

This study conducts two main experiments. The first experiment evaluates the basic capability of LLMs to independently correct OCR errors. The second experiment examines how incorporating a GNN, which captures the structural information of tables, improves restoration performance. By comparing the performance of the standalone LLM approach with the GNN-LLM hybrid approach, this study aims to analyze the importance of structural information in table data restoration.

4.1 Experiment 1: Can LLMs Independently Correct OCR Errors?

4.1.1 Objective and Necessity of the Experiment

OCR technology plays a crucial role in converting text from documents into digital data. However, it often introduces several issues. First, text loss and distortion frequently occur, such as truncated long sentences or reordered characters. Second, character misrecognition can lead to typographical errors, where visually similar characters like 'O' and '0' or 'B' and '8' are confused. Third, structural inconsistencies may arise, where multiple adjacent tables are mistakenly merged into one, or a single table is split into multiple parts, disrupting the original table structure.

Traditionally, OCR errors have been corrected manually or using rule-based algorithms. However, these methods require predefined rules tailored to specific datasets, limiting their generalizability. Recently, large language models (LLMs) have demonstrated outstanding performance in various text-based restoration tasks. Given their ability to handle structured text, they hold potential for correcting OCR errors in tabular data. The first experiment in this study aims to evaluate how effectively an LLM can independently compensate for OCR deficiencies without additional structural information.

4.1.2 Experimental Setup

In this experiment, we evaluate the performance of LLMs using GPT, Gemini, and Gemma models. The experimental dataset consists of OCR-error samples extracted from TabHD. The evaluation metrics include Precision, Recall, F1-score, and Table Accuracy, comparing the accuracy of the corrected data against the original OCR-extracted data.

4.1.3 Expected Results and Hypothesis

We hypothesize that LLMs will outperform rule-based OCR post-processing methods in correcting text errors. However, we also anticipate that LLMs will face limitations in fully restoring table structures, as they may struggle to consider structural relationships such as logical row-column connections and cell merging.

4.2 Experiment 2: Evaluating the Effectiveness of the Graph-Enhanced LLM Approach

4.2.1 Objective and Necessity of the Experiment

The first experiment confirmed that LLMs can partially correct OCR errors. However, it remains uncertain whether they can fully restore the structural integrity of tables. Unlike plain text data, tabular data requires maintaining the semantic relationships between rows and columns, necessitating an approach that goes beyond simple text-based methods.

Therefore, in the second experiment, we applied a method that converts table data into a graph structure and incorporates extracted graph information into the prompt. The primary objective of this study is to evaluate whether additional graph-based information contributes to enhancing the performance of LLMs.

4.2.2 Experimental Setup

The first experiment showed that the Gemini model performed best in correcting OCR errors. Therefore, in this second experiment, we focused on analyzing potential performance improvements based on Gemini. We compared three different models. First, we evaluated the OCR error correction performance of a standalone LLM model (Gemini). Second, we applied a fine-tuned LLM model (Finetuned Gemini) that had been trained on specific table restoration tasks, to assess the impact of domain-specific data training on performance. Third, we experimented with a graph-augmented Gemini model, where table data was converted into a graph structure, and the extracted graph information was added to the prompt. This experiment

aimed to determine whether leveraging graph information could improve performance (Tang et al., 2024).

The graphs used in the experiment contained structural relationships within tables and were incorporated into prompts to help the LLM better understand the context of tabular data. The experimental dataset was sampled from the table restoration tasks (merge, split, and cell recovery) in TabHD. The evaluation metrics used were Precision, Recall, and F1-score, consistent with those in Experiment 1.

4.2.3 Expected Results and Hypothesis

We hypothesize that the graph-augmented Gemini model will outperform the baseline LLM by better capturing the structural relationships within tables. Additionally, both the Finetuned Gemini and Graph-augmented Gemini models have undergone task-specific pretraining, which is expected to enable LLMs to utilize graph information more effectively. In particular, incorporating structural information into the prompt is expected to improve model comprehension in Table Merge and Table Split tasks.

However, graph information may not guarantee consistent performance improvements across all tasks, and in certain cases, it might act as redundant or unnecessary information. Another key factor influencing the experimental results will be how effectively the LLM can utilize graph-based information. Further research will be required to explore these aspects in depth.

5 Results

5.1 Evaluation Metrics

In this study, precision and F1-score were used as the primary performance metrics to evaluate the accuracy of OCR post-processing table restoration. The evaluation was designed to precisely reflect both the structural information and content of tables by applying a cell-level comparison approach rather than an individual word-level assessment.

Specifically, the restored tables generated by the model and the original tables were converted into a dataframe format, and precision and recall were calculated based on the match between individual cell values. Precision measures the proportion of correctly restored cells among those generated by the model, while recall represents the proportion of correctly recovered cells out of the total cells in the original table. Based on these two metrics, the F1-score was computed to assess the overall performance of the model.

5.2 Experiment 1: Can LLMs Independently Correct OCR Errors?

The experimental results are shown in Table 3. Overall, all three models exhibited relatively low performance, particularly struggling with restoring structurally complex tables. In the Table Merging task, the GPT model achieved the highest performance with a precision of 0.9249 and an F1-score of 0.6283. However, its performance dropped significantly in other tasks. In contrast, the Gemini and Gemma models demonstrated more balanced performance across tasks, but neither showed a high level of restoration capability.

In the Table Splitting task, all three models recorded low F1-scores. The GPT model, in particular, had a notably low F1-score of 0.1515, indicating that LLMs struggle to maintain structural consistency in table-splitting tasks. In the Table Cell Imputation task, all models achieved an F1-score in the range of 0.28 to 0.30, suggesting that missing data recovery remains incomplete.

For the Table Cell Correction task, three subtasks were evaluated: Fixing ambiguous characters, Restoring truncated text, and Restoring reordered text. All models demonstrated low overall performance. In the Fixing ambiguous characters subtask, GPT recorded a relatively higher F1-score of 0.3138. However, in the Restoring reordered text subtask, all models achieved an F1-score around 0.28, highlighting the difficulty of recovering text distorted by OCR errors.

When comparing the overall average performance, Gemini achieved the highest performance with a precision of 0.4187 and an F1-score of 0.3428, followed by Gemma with a precision of 0.4154 and an F1-score of 0.3371. The GPT model had the lowest performance, with a precision of 0.4038 and an F1-score of 0.3267. These results indicate that a purely LLM-based restoration approach is insufficient to fully correct OCR errors, emphasizing the necessity of additional modeling techniques that incorporate structural information.

5.3 Experiment 2: Evaluating the Effectiveness of the Graph-Enhanced LLM Approach

The experimental results are shown in Table 4. The Finetuned Gemini model demonstrated improved

Table 3: Performance comparison of Gemini, Gemma, and GPT models on various OCR-related tasks.

Task	Gemini		Gemma		GPT	
	Precision	F1-score	Precision	F1-score	Precision	F1-score
Table Merging	0.8990	0.6541	0.8982	0.6488	0.9249	0.6283
Table Splitting	0.3406	0.2885	0.3319	0.2776	0.3083	0.1515
Table Cell Imputation	0.3017	0.2852	0.3004	0.2821	0.3115	0.2767
Table Cell Correction (Fixing ambiguous characters)	0.2988	0.3040	0.3078	0.3091	0.2861	0.3138
Table Cell Correction (Restoring truncated text)	0.3869	0.3473	0.3668	0.3321	0.3226	0.3511
Table Cell Correction (Restoring reordered text)	0.2849	0.2740	0.2877	0.2728	0.2696	0.2887
Total Average	**0.4187**	**0.3428**	**0.4154**	**0.3371**	**0.4038**	**0.3267**

Table 4: Performance comparison of Gemini, Finetuned Gemini, and Graph-augmented Gemini models on various OCR-related tasks.

Task	Gemini		Finetuned Gemini		Graph-augmented Gemini	
	Precision	F1-score	Precision	F1-score	Precision	F1-score
Table Merging	0.8990	0.6541	0.9125	0.6692	0.8801	0.6204
Table Splitting	0.3406	0.2885	0.3582	0.3101	0.3156	0.2457
Table Cell Imputation	0.3017	0.2852	0.3204	0.3003	0.2898	0.2709
Table Cell Correction (Fixing ambiguous characters)	0.2988	0.3040	0.3078	0.3091	0.2964	0.3037
Table Cell Correction (Restoring truncated text)	0.3869	0.3473	0.3668	0.3321	0.3742	0.3458
Table Cell Correction (Restoring reordered text)	0.2849	0.2740	0.2877	0.2728	0.2815	0.2701
Total Average	**0.4187**	**0.3428**	**0.4422**	**0.3656**	**0.4063**	**0.3260**

performance in table structure-related tasks (Table Merging, Table Splitting, and Table Cell Imputation). In particular, in the Table Merging task, its precision increased to 0.9125 compared to the baseline Gemini model, and in the Table Splitting task, its performance also showed a slight improvement (F1-score 0.3101). These results suggest that pretraining on table structure recovery tasks contributed to enhancing the model's performance.

On the other hand, the Graph-augmented Gemini model exhibited an overall decrease in performance for table structure-related tasks. In the Table Merging task, its F1-score dropped to 0.6204, and a performance decline was also observed in the Table Splitting task (F1-score 0.2457). This indicates that additional graph information may introduce confusion rather than aiding the model's learning process.

Meanwhile, in the Cell Content Correction tasks (Fixing ambiguous characters, Restoring truncated text, and Restoring reordered text), all models performed similarly. Both the Finetuned Gemini and Graph-augmented Gemini models recorded nearly identical F1-scores to the baseline Gemini model, implying that neither graph information nor pretraining had a significant impact on text-based error correction tasks.

When comparing the overall average performance, the Finetuned Gemini model achieved the highest results, with a precision of 0.4422 and an F1-score of 0.3656. However, the Graph-augmented Gemini model showed a slight decrease in performance compared to the baseline Gemini model (Precision 0.4063, F1-score 0.3260). These results suggest that graph information may not consistently improve table structure recovery and, in some cases, may even hinder the model's predictions. Therefore, future research should focus on enhancing LLMs' ability to effectively utilize graph-based information to maximize its benefits in table restoration tasks.

6 Conclusion

In this study, we proposed TabHD, a large-scale benchmark dataset for restoring OCR-damaged table data, and evaluated LLM-based approaches. While traditional OCR post-processing focuses on text correction with limited table structure restoration, our experiments showed that LLMs can partially recover both table structures and cell contents.

The results indicated that standalone LLMs excel in text recovery but struggle with structural restoration. Fine-tuned LLMs improved performance in tasks like Table Merging and Table Splitting. We also tested a Graph-augmented LLM approach, in-

corporating table structure as graph-based prompts, but observed inconsistent improvements, highlighting the need for better graph utilization.

7 Limitations

While this study represents a significant attempt to explore the potential of LLMs for table restoration in OCR post-processing, several limitations remain. First, the benchmark dataset TabHD, constructed in this study, reflects various OCR errors but does not encompass all possible transformations present in real-world data. Therefore, future research should focus on expanding the dataset to include a wider range of domains and layouts.

Second, the proposed methods rely on computationally expensive LLMs, which necessitate optimization strategies for real-time OCR post-processing applications. Developing lightweight models or methodologies that minimize latency will be an important direction for future research.

To address these limitations, future studies will consider developing specialized models trained on OCR error patterns, exploring effective methods for utilizing graph-based information, expanding the benchmark dataset to include diverse domain data, and designing more efficient LLM architectures.

References

Nora Abdelmageed, Sirko Schindler, and Birgitta König-Ries. 2021. Biodivtab: A table annotation benchmark based on biodiversity research data. In *SemTab@ ISWC*, pages 13–18.

Salvatore Carta, Alessandro Giuliani, Leonardo Piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative zero-shot llm prompting for knowledge graph construction. *arXiv preprint arXiv:2307.01128*.

Madelon Hulsebos, Çagatay Demiralp, and Paul Groth. 2023. Gittables: A large-scale corpus of relational tables. *Proceedings of the ACM on Management of Data*, 1(1):1–17.

Zheyuan Liu, Xiaoxin He, Yijun Tian, and Nitesh V Chawla. 2024. Can we soft prompt llms for graph learning tasks? In *Companion Proceedings of the ACM on Web Conference 2024*, pages 481–484.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of EMNLP*.

Chirag Patel, Atul Patel, and Dharmendra Patel. 2012. Optical character recognition by open source ocr tool tesseract: A case study. *International journal of computer applications*, 55(10):50–56.

ShengYun Peng, Seongmin Lee, Xiaojing Wang, Rajarajeswari Balasubramaniyan, and Duen Horng Chau. 2024. Unitable: Towards a unified framework for table structure recognition via self-supervised pretraining. *arXiv preprint arXiv:2403.04822*.

Alexey Shigarov. 2023. Table understanding: Problem overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(1):e1482.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024a. 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, page 645–654, New York, NY, USA. Association for Computing Machinery.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024b. 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*, pages 645–654.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491–500.

Kai Wang, Yuwei Xu, Zhiyong Wu, and Siqiang Luo. 2024a. Llm as prompter: Low-resource inductive reasoning on arbitrary knowledge graphs. *arXiv preprint arXiv:2402.11804*.

Yu Wang, Nedim Lipka, Ryan A Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2024b. Knowledge graph prompting for multi-document question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19206–19214.

Liangyu Zha, Junlin Zhou, Liyao Li, Rui Wang, Qingyi Huang, Saisai Yang, Jing Yuan, Changbao Su, Xiang Li, Aofeng Su, et al. 2023. Tablegpt: Towards unifying tables, nature language and commands into one gpt. *arXiv preprint arXiv:2307.08674*.

Jiawei Zhang. 2023. Graph-toolformer: To empower llms with graph reasoning ability via prompt augmented by chatgpt. *arXiv preprint arXiv:2304.11116*.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2023. Tablellama: Towards open large generalist models for tables. *arXiv preprint arXiv:2311.09206*.