TabFill: Sketch-Guided Missing Value Imputation on Tabular Data via Large Language Models

Changlun Li
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou City, China
cli942@connect.hkust-gz.edu.cn

ABSTRACT

Data preprocessing plays a pivotal role in data-driven analysis, with Missing Value Imputation (MVI) standing out as a crucial and time-consuming task. Despite extensive efforts devoted to MVI, existing approaches often fall short in accuracy, relying on simple methods such as descriptive statistics like mean, mode, or zero imputation. In this paper, we present TabFill, a novel approach to archive MVI through the power of Large Language Models (LLMs). TabFill brings up a combination strategy: sketch and code generation. Firstly, user can sketch a series of planning steps to guide LLM in providing answers with explanations. Later, those results would be constrained into a structured Python code via CodeLLM. This approach not only improves the reliability and efficiency of imputations but also facilitates human review by presenting clear, actionable code. Through empirical evaluations on diverse datasets, we demonstrate that TabFill significantly enhances MVI efficiency and maintain good accuracy, offering a promising advancement in data preprocessing techniques for data-driven analyses. We have released codes in Github: https://github.com/tigerlcl/llm-sketch.

KEYWORDS

Data preparation, Missing value imputation, Large language model

ACM Reference Format:
Changlun Li. 2024. TabFill: Sketch-Guided Missing Value Imputation on Tabular Data via Large Language Models. In . ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In the era of data-driven decision-making, the quality of the data being analyzed is paramount. As the adage goes, "rubbish in, rubbish out." This highlights the importance of data preprocessing, particularly Missing Value Imputation (MVI), which ensures that machine learning models are fed with high-quality datasets. Without proper handling of missing data, the predictive ability of these models can be significantly compromised due to the contamination from incomplete records. Missing data is a ubiquitous problem across various domains, from healthcare to finance [1, 2]. Studies have shown that in real-world datasets, the prevalence of missing data can range from minor omissions to substantial portions of the dataset, often exceeding 30% in certain cases [2]. This presents a substantial challenge, as inaccurate handling of these gaps can lead to biased and unreliable analytical outcomes. Existing MVI techniques come with several notable limitations:

- Out-dated approach: Methods such as mean, mode, or hot-deck imputation [1, 5] often lack the accuracy needed for high-quality data analysis, as shown in Figure 1(a). These techniques can still leave data in a dirty state or necessitate the removal of incomplete records. Besides, traditional methods fail to leverage the context provided by existing entries in the dataset or domain-specific knowledge, resulting in suboptimal imputation.
- Human-aided overhead: Data scientists reportedly spend up to 80% of their project time on data cleaning tasks [2], which includes MVI. This high level of manual effort is not scalable and can lead to inconsistencies and errors. The reliance on human intervention for imputing missing values is labor-intensive and time-consuming, diverting valuable resources away from more strategic analytical tasks.

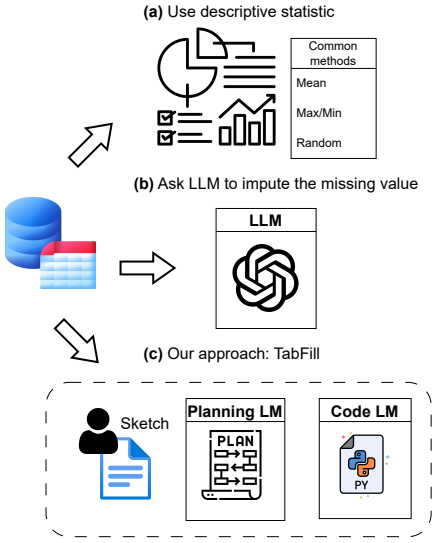

Figure 1: Imputation using different approaches

In recent years, large language models (LLMs) have shown considerable promise in addressing complex data processing tasks, thanks to their profound capabilities [21], table understanding [4, 13, 20] in knowledge extraction and content generation. Their ability

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

to understand and manipulate textual and, increasingly, non-textual data suggests a promising avenue for enhancing MVI techniques. Therefore, we explored two existing LLM approaches: 1) instruct LLM directly to perform MVI on dirty data (see Figure 1(b)); 2) employ Chain-of-Thought prompting [18] to enhance their reasoning process on MVI. However, Challenges were revealed by the LLM output due to unreliable and extensive explanations. LLM exhibits insufficient filling rate when confronted with numerous missing values.

To address above challenges, we propose TabFill (see Figure 1(c)), a novel framework that combines sketch-based guidance, planning and code generation using LLMs to archive MVI. Initially, users sketch a series of steps as instructions, then it will guide the LLM in imputation planning with proper explanations. These results are then translated into executable Python code using a Code LLM, enhancing both the reliability and efficiency of the imputation process. This method not only automates the imputation process but also ensures that the outputs are clear and actionable for human reviewers. To summerize, our contributions of this paper are the followings:

- Sketch-guided context awareness: By composing a domain-specific sketch, TabFill can follow the guidance to generate imputation planning. By harnessing the domain knowledge embedded in LLMs, its imputation strategy would take the context of dataset into consideration, compared to traditional methods.
- Structured output: TabFill offers a more explainable and portable output in the end. The imputation plan will be converted into Python code with the integration of Code LLM. Such an executable output can be validated and modified as needed, providing transparency and flexibility.

In the following sections, we will introduce some related work and the methodology behind TabFill, present empirical evaluations on diverse datasets, and discuss the implications of our findings.

2 RELATED WORK

Missing Value Imputation (MVI) has been explored with the advent of data science, aiming to look at the patterns and mechanisms that create the missing data, as well as a taxonomy of missing data in datasets [5]. Despite a single imputation method like Hot-Deck [1] which is imputed from a randomly selected similar record, we categorize existing solutions in the context of statistics and machine intelligence for missing value imputation as follows.

Statistic-based Methods. Initially, the most common strategy for MVI is using a descriptive statistic, e.g., mean, median, or most frequent, along each column, or using a constant value. It is widely adopted in existing packages and tools, such as *sklearn SimpleImputer* [7] and *Excel Power Query*. Besides, curated packages such as MICE [17] can impute incomplete multivariate data by chained equations.

Machine Learning-based Methods. Razavi-Far et al. [9] propose kEMI and kEMI+ for imputing categorical and numerical missing data correspondingly. They both first utilize the k-nearest neighbours (KNN) algorithm to search the K-top similar records to a record with missing values, then invoke the Expectation-Maximization Imputation (EMI) algorithm, which uses the feature correlation among the K-top similar records to impute missing values. Tao et al. [14] improve algorithms in Reverse kNN, allowing to retrieve an arbitrary number of neighbours in multiple dimensions. Sridevi et al. [11] propose ARLSimpute, an autoregressive model to predict missing values. Stekhoven and Bühlmann [12] propose an iterative imputation method based on the random forest, the missForest, that can impute the missing value of mixed-type data. Tsai et al. [16] propose CCMVI, which calculates the distances between observed data and the class centres to define the threshold for later imputation.

Deep Learning-based Methods. Gondara and Wang [3] propose a multiple imputation model based on overcomplete deep denoising autoencoders, which is capable of handling different missing situations in terms of the data types, patterns, proportions, and distributions. Zheng and Charoenphakdee [22] explore the use of Conditional Score-based Diffusion Models for Tabular data (TabCSDI) to impute missing values in tabular datasets. Their study evaluates three techniques for effectively handling categorical variables and numerical variables simultaneously. Specifically, the CSDI [15] also acts as a time series imputation method that utilizes score-based diffusion models to exploit correlations on observed values.

LLM-based Methods. With the advance of LLM, especially text generation models such as GPT-4, some techniques can be applied to MVI task. Since LLMs are trained on extensive and diverse corpora, they inherently possess knowledge of a wide array of common entities [6, 10], intuitively, we can directly ask LLM to perform MVI given some dirty data. Besides, CoT prompting [18] can significantly improve the ability of complex reasoning. Alternatively, Poldrack et al. [8] explore the code generation ability utilizing LLM so that a specific script for MVI can be generated.

3 METHODOLOGY

3.1 Overview

TabFill is a novel framework that combines sketch-based guidance, planning, and code generation using Large Language Models (LLMs) to achieve Missing Value Imputation (MVI). The process begins with the user providing an initial message to fix the table. The system follows several steps to ensure reliable and efficient imputation (See Figure 2):

(1) User Initial Message: The process starts when the user sends a message indicating the need for imputation, such as "help fix table".
(2) Planning LM: Following a predefined sketch, the Planning LM generates an imputation plan, providing pseudo-code instructions.
(3) Code Generation and Execution: The plan is converted into executable Python code by a Code Generation LM, designed with prompts to treat it as a code writer. Later, the generated code is validated against ground truth data to verify the accuracy of the imputation plan.
(4) Evaluator: The executed output is evaluated, and if the imputation result is not identical to the ground truth value, the system allows retries. If the code is correct, it proceeds to

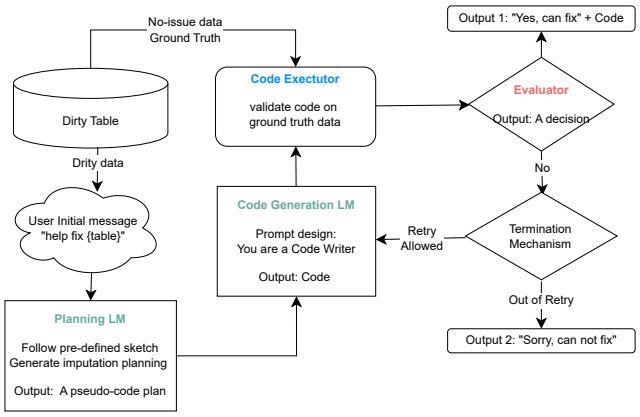

Figure 2: An overview of TabFill

the final output; otherwise, it terminates the process after retries are exhausted.

By integrating these components, TabFill provides a comprehensive and efficient solution for MVI, ensuring that the imputation process is not only automated but also transparent and verifiable. This structured approach reduces human intervention to the initial and final stages, thereby optimizing both the reliability and scalability of the MVI tasks.

3.2 Sketch-Guided Planning

The sketch serves as a critical component in guiding the LLM for MVI tasks. Inspired by the instruction-following feature of LLMs [23], the sketch is predefined and specific to MVI on tabular data. It acts as a step-by-step instruction set for the LLM to plan the imputation steps. The predefined sketch in our later experiment includes: 1) identifying columns with missing values; 2) finding related columns that can help derive the missing values; 3) generating an imputation solution; 4) applying the solution; This structured approach leverages the domain knowledge embedded in LLMs, allowing for context-aware imputation strategies that outperform traditional methods.

3.3 Code Generation

As Inspired by previous research [8], the Code Generation LM bridges the gap between textual planning and code implementation. Given the importance of efficiency in MVI, converting imputation plans into executable scripts is a strategic choice. In our experiment, Python code is used to implement the MVI solutions, ensuring that the imputation process is not only automated but also easily verifiable and modifiable. This step enhances the transparency and portability of the imputation outcomes.

3.4 Evaluator

The evaluator acts as a decision-maker, crucial for validating the imputation results. It compares the executed output with ground truth data to assess the correctness of the current coding solution. If discrepancies are found, the system allows for retries to refine the code. This iterative validation process ensures high accuracy and reliability in the final imputed data.

Overall, TabFill's innovative framework provides a robust, automated approach to MVI, leveraging the strengths of LLMs to deliver reliable and efficient solutions that significantly reduce the need for manual intervention.

4 EXPERIMENT

4.1 Setup

Approaches. We compared TabFill with the following approaches: (1) *LLM:* It utilizes LLM intuitively without any instructions. (2) *CoT:* It prompts LLM to reply with "Let's think step by step".

Backend LLM.. We employ OpenAI's latest model, GPT-4-Turbo,[1] as the backend model. The GPT request is supported by API service from the Internet. The token size is set as 4096 using a temperature of 0. Specific to planning and Code LLM, they are both initialized by prompting techniques. In order to develop a Python executable runtime, we adopted the AutoGen framework [19], which builds an interactive environment for agentic LLM. The code writer and code executor will complete the MVI job in automatic manner given a maximum retry limits.

Evaluation Metric. We assess overall accuracy by comparing imputed values to ground truth values using the exact match technique, and then counting the entire amount.

4.2 Dataset

In our experiments, we utilized two reprentative datasets from Kaggle to evaluate the performance of TabFill:

Table 1: Fact of dataset

Dataset	Flight	Supermarket
# of attributes	9	6
# of tuples per col	5	5
Testing variables	Source, Total_Stops	Tax 5%, Total

Flight Price Prediction Dataset (Flight): This dataset covers the historical airline flight of India during a certain period. It includes features such as source, destination, route, departure and arrival times, duration, and total stops. The dataset is used to predict flight prices based on these features. It offers a rich set of attributes with potential missing values that require imputation for accurate predictive modeling. In this dataset, we aim to backfill the Source and Total_Stops fields (See Appendix 3). Based on the context information, we can conclude the relationship as below:

- Source is equivalent to the first stop of Route, where the stop code should be translated to city name. Let us use Route: MAA → CCU for instance, the missing value of Source is translated from "MAA", which is Chennai.
- Total_Stops is equal to the number of arrows in Route, then minus one. For example, the missing value should be "1 stop" for the Route: DEL → HYD → COK.

[1]The newer model, gpt-4o was not released yet at the time of experiment

Supermarket Sales Dataset (Supermarket): The dataset is originally sourced from the 3-month historical sales transaction of a supermarket company in Myanmar. It includes features such as unit price, quantity, tax, total, and payment. The dataset is used to analyze and predict sales performance, requiring imputation of missing values to ensure comprehensive and accurate analysis. In this dataset, we aim to backfill the Tax 5% and Total fields (See Appendix 4). Based on the context information, we can conclude the relationship as below:

- $Tax\ 5\% = 5\% * (Unit\ price * Quantity)$, the special percentage in column name indicates the tax rate, thus, the missing value at the row 1 should be 33.25.
- $Total = (Unit\ price * Quantity) + Tax\ 5$, the total spending is tax included, thus, the missing value in row 4 is 19.95.

Notably, the *Flight* dataset stands for logical-based associations between attributes. While, the *Supermarket* dataset stands for formula-based associations.

Preprocessing. Each dataset is randomly segmented into subsets to accommodate the imputation of missing values that require proper strategies. Given the transparent association of the target variable to be imputed and other known variables, the subset is well curated as a testing tuple. Specifically, we mask the original values of the target variable with `NaN` to mimic the missing value. We run a few experiments for demonstration purpose only. Full statistics are shown in Table 1.

4.3 Result Analysis

In this section, we report the performance of MVI using different approaches on two datasets. Please refer to the Table 2. Next, we will analyze the experimental results by dataset.

Table 2: Experiment result using different approaches

	Flight		Supermarket	
Approach	Source	Total_Stops	Tax 5%	Total
LLM	3	3	5	5
CoT	**4**	**4**	**5**	**5**
TabFill	**4**	2	**5**	4

Flight:

- LLM Approach: The basic LLM approach achieved a score of 3 for both the "Source" and "Total_Stops" columns. This indicates a moderate level of accuracy but highlights potential limitations in handling more complex imputation tasks within the dataset.
- CoT Approach: This method shows an improvement over the basic LLM, achieving a score of 4 in both columns. The CoT approach benefits from its structured reasoning process, allowing for better context understanding and more accurate imputations.
- TabFill Approach: Our proposed TabFill framework matches the CoT approach in the "Source" column with a score of 4 but excels in the "Total_Stops" column with a score of 2, indicating a lower accuracy. Because LLM may get confused by the representation learning of flight route, in particularly with symbols.

Below is a sample code snippet that TabFill generated eventually for MVI:

```python
# flight_2.py
def impute_total_stops(route):
    stops_count = route.count('->') - 1
    if stops_count == 0:
        return 'non-stop'
    elif stops_count == 1:
        return '1 stop'
    elif stops_count == 2:
        return '2 stops'
    else:
        return f'{stops_count} stops'

# flight_7.py
def impute_missing_source(route):
    if route:
        return route.split("->")[0]
    return None
```

Listing 1: Python function to impute missing values in Flight

Supermarket:

- LLM Approach: The basic LLM approach performed consistently with scores of 5 in both the "Tax 5%" and "Total" columns. This suggests challenges in achieving accurate imputations for the given attributes.
- CoT Approach: Similar to the LLM approach, the CoT method also scored 5 in both columns. While it improves reasoning, it did not significantly enhance the accuracy in this dataset.
- TabFill Approach: TabFill maintained a perfect score of 5 for the "Tax 5%" column, comparable to other methods, but demonstrated an improvement in the "Total" column with a score of 4. This improvement underscores TabFill's ability to deliver more accurate and contextually aware imputations, facilitated by the structured output and validation mechanisms inherent in our framework.

Below is a sample code snippet that TabFill generated eventually for MVI:

```python
# supermarket_9.py
def calculate_tax(unit_price, quantity):
    return (unit_price * quantity) * 0.05

# supermarket_8.py
def calculate_total(unit_price, quantity, tax):
    return (unit_price * quantity) + tax
```

Listing 2: Python function to impute missing values in Supermarket

Overall, TabFill exhibits a notable improvement in performance compared to other methods, particularly in the logical-associated columns where understanding the context is crucial. The integration of sketch-guided planning and code generation ensures that the imputation process is not only more accurate but also more

efficient and scalable. This validation against ground truth data and iterative refinement further solidifies TabFill as a robust solution for missing value imputation in diverse datasets.

5 CONCLUSION

In this paper, we introduced TabFill, a novel framework for Missing Value Imputation (MVI) that leverages the capabilities of Large Language Models (LLMs) through a combination of sketch-based guidance, planning, and code generation. Our approach significantly enhances the efficiency of imputation tasks, reducing the reliance on manual intervention and providing a structured, executable output that is both transparent and flexible.

The results from our experiments on diverse datasets, including flight route and supermarket sales, demonstrate the effectiveness of TabFill. Our method not only outperforms traditional imputation techniques but also shows great performance compared to other LLM-based approaches. TabFill's ability to generate context-aware imputations and convert them into verifiable Python code ensures robust and reliable data preprocessing.

Contributed to the framework design, the need for human input is minimized to the initial sketching and final code review stages, streamlining the MVI process. This approach not only reduces the time and effort typically spent on manual imputation but also mitigates the risk of human error.

6 LIMITATION

While TabFill demonstrates significant improvements in Missing Value Imputation (MVI), it has certain limitations that need to be addressed in future work:

- Dependency on Initial Sketch: The effectiveness of TabFill heavily relies on the quality and completeness of the initial sketch provided by the user. Inadequate or poorly defined sketches can lead to suboptimal imputation plans.
- Computational Cost: As API models are paid services, the computational cost associated with using LLMs can become a significant factor, especially for large-scale imputation tasks. This cost needs to be carefully managed and optimized to ensure the approach remains cost-effective.
- Handling Highly Correlated Missing Data: The current approach may struggle with datasets where missing values are highly correlated across multiple columns and rows, requiring more sophisticated techniques to accurately impute these values.

Future work will focus on refining the initial sketching process, exploring the use case for open-source model as backend, and validating across a wider array of datasets and domains. Additionally, integrating advanced techniques, such as LLM agent to handle highly correlated missing data will be a priority, aiming to further enhance the robustness and applicability of TabFill.

REFERENCES

[1] Rebecca R Andridge and Roderick JA Little. 2010. A review of hot deck imputation for survey non-response. *International statistical review* 78, 1 (2010), 40–64.

[2] Tlamelo Emmanuel, Thabiso Maupong, Dimane Mpoeleng, Thabo Semong, Banyatsang Mphago, and Oteng Tabona. 2021. A survey on missing data in machine learning. *Journal of Big data* 8 (2021), 1–37.

[3] Lovedeep Gondara and Ke Wang. 2018. Mida: Multiple imputation using denoising autoencoders. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part III 22*. Springer, 260–272.

[4] Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. 2023. Table-gpt: Table-tuned gpt for diverse table tasks. *arXiv preprint arXiv:2310.09263* (2023).

[5] R.J.A. Little and D.B. Rubin. 2019. *Statistical Analysis with Missing Data*. Wiley. https://books.google.com/books?id=BemMDwAAQBAJ

[6] Avanika Narayan, Ines Chami, Laurel Orr, and Christopher Ré. 2022. Can Foundation Models Wrangle Your Data? *Proceedings of the VLDB Endowment* 16, 4 (2022), 738–746.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[8] Russell A Poldrack, Thomas Lu, and Gašper Beguš. 2023. AI-assisted coding: Experiments with GPT-4. *arXiv preprint arXiv:2304.13187* (2023).

[9] Roozbeh Razavi-Far, Boyuan Cheng, Mehrdad Saif, and Majid Ahmadi. 2020. Similarity-learning information-fusion schemes for missing data imputation. *Knowledge-Based Systems* 187 (2020), 104805.

[10] Simon Razniewski, Andrew Yates, Nora Kassner, and Gerhard Weikum. 2021. Language models as or for knowledge bases. *arXiv preprint arXiv:2110.04888* (2021).

[11] S. Sridevi, S. Rajaram, C. Parthiban, S. SibiArasan, and C. Swadhikar. 2011. Imputation for the analysis of missing values and prediction of time series data. In *2011 International Conference on Recent Trends in Information Technology (ICRTIT)*. 1158–1163. https://doi.org/10.1109/ICRTIT.2011.5972466

[12] Daniel J Stekhoven and Peter Bühlmann. 2012. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics* 28, 1 (2012), 112–118.

[13] 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*. 645–654.

[14] Yufei Tao, Dimitris Papadias, and Xiang Lian. 2004. Reverse knn search in arbitrary dimensionality. In *Proceedings of the Very Large Data Bases Conference (VLDB), Toronto*.

[15] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. 2021. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems* 34 (2021), 24804–24816.

[16] Chih-Fong Tsai, Miao-Ling Li, and Wei-Chao Lin. 2018. A class center based approach for missing value imputation. *Knowledge-Based Systems* 151 (2018), 124–135.

[17] Stef Van Buuren and Karin Groothuis-Oudshoorn. 2011. mice: Multivariate imputation by chained equations in R. *Journal of statistical software* 45 (2011), 1–67.

[18] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs.CL]

[19] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation Framework. arXiv:2308.08155 [cs.AI]

[20] Liangyu Zha, Junlin Zhou, Liyao Li, Rui Wang, Qingyi Huang, Saisai Yang, Jing Yuan, Changbao Su, Xiang Li, Aofeng Su, Tao Zhang, Chen Zhou, Kaizhe Shou, Miao Wang, Wufang Zhu, Guoshan Lu, Chao Ye, Yali Ye, Wentao Ye, Yiming Zhang, Xinglong Deng, Jie Xu, Haobo Wang, Gang Chen, and Junbo Zhao. 2023. TableGPT: Towards Unifying Tables, Nature Language and Commands into One GPT. arXiv:2307.08674 [cs.AI]

[21] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. arXiv:2303.18223 [cs.CL]

[22] Shuhan Zheng and Nontawat Charoenphakdee. 2022. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128* (2022).

[23] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911* (2023).

Table 3: Flight dirty demo

Airline	Date_of_Journey	Source	Destination	Route	Total_Stops
IndiGo	12/06/2019	Chennai	Kolkata	MAA → CCU	non-stop
Jet Airways	3/06/2019	Mumbai	Hyderabad	BOM → HYD	non-stop
Jet Airways	15/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	2 stops
Jet Airways	21/03/2019	Delhi	Cochin	DEL → IXC → BOM → COK	2 stops
Vistara	15/03/2019	**NaN**	Kolkata	MAA → CCU	non-stop
IndiGo	3/03/2019	Delhi	Cochin	DEL → HYD → COK	**NaN**
Air Asia	24/03/2019	Kolkata	Banglore	CCU → DEL → BLR	1 stop

Table 4: Supermarket dirty demo

Product line	Unit price	Quantity	Tax 5%	Total	Payment
Home and lifestyle	95	7	**NaN**	698.25	Credit card
Home and lifestyle	24	7	8.4	176.4	Ewallet
Home and lifestyle	94	8	37.6	789.6	Ewallet
Health and beauty	19	1	0.95	**NaN**	Credit card
Home and lifestyle	49	10	24.5	514.5	Credit card