Large Language Models on Tabular Data - A Survey

Anonymous authors
Paper under double-blind review

Abstract

Recent breakthroughs in large language modeling have facilitated rigorous exploration of their application in diverse tasks related to tabular data modeling, such as prediction, tabular data synthesis, question answering, and table understanding. Each task presents unique challenges and opportunities. However, there is currently a lack of comprehensive review that summarizes and compares the key techniques, metrics, datasets, models, and optimization approaches in this research domain. This survey aims to address this gap by consolidating recent progress in these areas, offering a thorough survey and taxonomy of the datasets, metrics, and methodologies utilized. It identifies strengths, limitations, unexplored territories, and gaps in the existing literature, while providing some insights for future research directions in this vital and rapidly evolving field. It also provides relevant code and datasets references. Through this comprehensive review, we hope to provide interested readers with pertinent references and insightful perspectives, empowering them with the necessary tools and knowledge to effectively navigate and address the prevailing challenges in the field.

1 Introduction

Large language models (LLMs) are deep learning models trained on extensive data, endowing them with versatile problem-solving capabilities that extend far beyond the realm of natural language processing (NLP) tasks (Fu & Khot, 2022). Recent research has revealed emergent abilities of LLMs, such as improved performance on few-shot prompted tasks (Wei et al., 2022b). The remarkable performance of LLMs have incited interest in both academia and industry, raising beliefs that they could serve as the foundation for Artificial General Intelligence (AGI) of this era (Chang et al., 2024; Zhao et al., 2023b; Wei et al., 2022b). A noteworthy example is ChatGPT, designed specifically for engaging in human conversation, that demonstrates the ability to comprehend and generate human language text (Liu et al., 2023g).

Before LLMs, researchers have been investigating ways to integrate tabular data with neural network for NLP and data management tasks (Badaro et al., 2023). Today, researchers are keen to investigate the abilities of LLMs when working with tabular data for various tasks, such as prediction, table understanding, quantitative reasoning, and data generation (Hegselmann et al., 2023; Sui et al., 2023c; Borisov et al., 2023a).

Tabular data stands as one of the pervasive and essential data formats in machine learning (ML), with widespread applications across diverse domains such as finance, medicine, business, agriculture, education, and other sectors that heavily rely on relational databases (Sahakyan et al., 2021; Rundo et al., 2019; Hernandez et al., 2022; Umer et al., 2019; Luan & Tsai, 2021). Tabular data, commonly known as structured data, refers to data organized into rows and columns, where each column represents a specific feature. In this section, we first introduce the characteristics of tabular data, then provide a brief review of traditional, deep-learning and LLM methods tailored for this area. At last, we articulate the contribution of the paper and provide a layout of the following sections.

1.1 Characteristics of tabular data

This subsection discusses the unique characteristics and challenges posed by tabular data:

1. Heterogeneity: Tabular data can contain different feature types: categorical, numerical, binary, and textual. Therefore, features can range from being dense numerical features to sparse or high-cardinality categorical features (Borisov et al., 2022).

2. Sparsity: Real-world applications, such as clinical trials, epidemiological research, fraud detection, etc., often deal with imbalanced class labels and missing values, which results in long-tailed distribution in the training samples (Sauber-Cole & Khoshgoftaar, 2022).

3. Dependency on pre-processing: Data pre-processing is crucial and application-dependent when working with tabular data. For numerical values, common techniques include data normalization or scaling, categorical value encoding, missing value imputation, and outlier removal. For categorical values, common techniques include label encoding or one-hot encoding. Improper pre-processing may lead to information loss, sparse matrix, and introduce multi-collinearity (e.g. with one-hot encoding) or synthetic ordering (e.g. with ordinal encoding) (Borisov et al., 2023a).

4. Context-based interconnection: In tabular data, features can be correlated. For example, age, education, and alcohol consumption from a demographic table are interconnected: it is hard to get a doctoral degree at a young age, and there is a minimum legal drinking age. Including correlated regressors in regressions lead to biased coefficients, hence, a modeler must be aware of such intricacies (Liu et al., 2023d).

5. Order invariant: In tabular data, examples can be sorted. However, as opposed to text-based and image-based data that is intrinsically tied to the position of the word/token or pixel in the text or image, tabular examples are relatively order-invariant. Therefore, position-based methodologies (e.g., spatial correlation, impeding inductive bias, convolutional neural networks (CNN)) are less applicable for tabular data modeling (Borisov et al., 2022).

6. Lack of prior knowledge: In image or audio data, there is often prior knowledge about the spatial or temporal structure of the data, which can be leveraged by the model during training. However, in tabular data, such prior knowledge is often lacking, making it challenging for the model to understand the inherent relationships between features (Borisov et al., 2022; 2023a).

1.2 Traditional and deep learning in tabular data

Traditional tree-based ensemble methods such as gradient-boosted decision trees (GBDT) remain the state-of-the-art (SOTA) for predictions on tabular data (Borisov et al., 2022; Gorishniy et al., 2021)). In boosting ensemble methods, base learners are learned sequentially to reduce previous learner's error until no significant improvement are made, making it relatively stable and accurate than a single learner (Chen & Guestrin, 2016). Traditional tree-based models are known for its high performance, efficiency in training, ease of tuning, and ease of interpretation. However, they have limitations compared to deep learning models: 1. Tree-based models can be sensitive to feature engineering especially with categorical features while deep learning can learn representation implicitly during training (Goodfellow et al., 2016). 2. Tree-based models are not naturally suited for processing sequential data, such as time series while deep learning models such as Recurrent Neural Networks (RNNs) and transformers excel in handling sequential dependencies. 3. Tree-based models sometimes struggle to generalize to unseen data particularly if the training data is not representative of the entire distribution, while deep learning methods may generalize better to diverse datasets with their ability to learn intricate representations (Goodfellow et al., 2016).

In the recent years, many works have delved into using deep learning for tabular data modeling. The methodologies can be broadly grouped into the following categories: 1. Data transformation. These models either strive to convert heterogenous tabular input into homogenous data more suitable to neural networks, like an image, on which CNN-like mechanism can be applied (SuperTML (Sun et al., 2019), IGTD (Zhu et al., 2021b), 1D-CNN (Kiranyaz et al., 2019)), or methods focusing on combining feature transformation with deep neural networks (Wide&Deep (Cheng et al., 2016; Guo & Berkhahn, 2016), DeepFM (Guo et al., 2017), DNN2LR (Liu et al., 2021)). 2. Differentiable trees. Inspired by the performance of ensembled trees, this line of methods seeks to make trees differentiable by smoothing the decision function (NODE (Popov et al., 2019), SDTR (Luo et al., 2021), Net-DNF (Katzir et al., 2020)). Another subcategory of methods combine tree-based models with deep neural networks, thus can maintain tree's capabilities on handling sparse categorical features (DeepGBM (Ke et al., 2019a)), borrow prior structural knowledge from the tree (TabNN (Ke et al., 2019b)), or exploit topological information by converting structured data into a directed graph (BGNN (Ivanov & Prokhorenkova, 2021). 3. Attention-based methods. These models incorporate attention mechanisms for feature selection and reasoning (TabNet (Arik & Pfister, 2020)), feature encoding (TransTab (Wang & Sun, 2022), TabTransformer (Huang et al., 2020)), feature interaction modeling (ARM-net (Cai et al., 2021)), or aiding intrasample information sharing (SAINT (Somepalli et al., 2021), NPT (Kossen et al., 2022)). 4. Regularization methods. The importance of features varies in tabular data, in contrast to image or text data. Thus, this line of research seeks to design an optimal and dynamic regularization mechanism to adjust the sensitivity of the model to certain inputs (e.g. RLN (Shavitt & Segal, 2018), Regularization Cocktails (Kadra et al., 2021). In spite of rigorous attempts in applying deep learning to tabular data modeling, GBDT algorithms, including XGBoost, LightGBM, and CatBoost (Prokhorenkova et al., 2019), still outperform deep-learning methods in most datasets with additional benefits in fast training time, high interpretability, and easy optimization (Shwartz-Ziv & Armon, 2022; Gorishniy et al., 2021; Grinsztajn et al., 2022). Deep learning models, however, may have their advantages over traditional methods in some circumstances, for example, when facing very large datasets, or when the data is primarily comprised of categorical features (Borisov et al., 2022).

Another important task for tabular data modeling is data synthesis. Abilities to synthesize real and high-quality data is essential for model development. Data generation is used for augmentation when the data is sparse (Onishi & Meguro, 2023), imputing missing values (Jolicoeur-Martineau et al., 2023), and class rebalancing in imbalanced data (Sauber-Cole & Khoshgoftaar, 2022). Traditional methods for synthetic data generation are mostly based on Copulas (Patki et al., 2016; Li et al., 2020) and Bayesian networks (Zhang et al., 2017) while recent advancement in generative models such as Variational Autoencoders (VAEs) (Ma et al., 2020; Darabi & Elor, 2021; Vardhan & Kok, 2020; Liu et al., 2023d)), generative adversarial networks (GANs) (Park et al., 2018; Choi et al., 2018; Baowaly et al., 2019; Xu et al., 2019), diffusion (Kotelnikov et al., 2022; Xu et al., 2023; Kim et al., 2022b;a; Lee et al., 2023; Zhang et al., 2023c), and LLMs, opened up many new opportunities. These deep learning approaches have demonstrated superior performance over classical methods such as Bayesian networks ((Xu et al., 2019)).

Table question answering (QA) is a natural language research problem from tabular data. Many earlier methods fine-tune BERT (Devlin et al., 2019) to become table encoders for table-related tasks, like TAPAS (Herzig et al., 2020), TABERT (Yin et al., 2020b), TURL (Deng et al., 2022a), TUTA (Wang et al., 2021) and TABBIE (Iida et al., 2021). For example, TAPAS extended BERT's masked language model objective to structured data by incorporating additional embeddings designed to capture tabular structure. It also integrates two classification layers to facilitate the selection of cells and predict the corresponding aggregation operator. A particular table QA task, Text2SQL, involves translating natural language question into structured query language (SQL). Earlier research conducted semantic parsing through hand-crafted features and grammar rules (Pasupat & Liang, 2015b). Semantic parsing is also used when the table is not coming from non-database tables such as web tables, spreadsheet tables, and others (Jin et al., 2022). Seq2SQL is a sequence-to-sequence deep neural network using reinforcement-learning to generate conditions of query on WikiSQL task (Zhong et al., 2017a). Some methodologies are sketch-based, wherein a natural language question is translated into a sketch. Subsequently, programming language techniques such as type-directed sketch completion and automatic repair are utilized in an iterative manner to refine the initial sketch, ultimately producing the final query (e.g. SQLizer (Yaghmazadeh et al., 2017)). Another example is SQLNet (Xu et al., 2017) which uses column attention mechanism to synthesize the query based on a dependency graph-dependent sketch. A derivative of SQLNet is TYPESQL (Yu et al., 2018a) which is also a sketch-based and slot-filling method entails extracting essential features to populate their respective slots. Unlike the previous supervised end-to-end models, TableQuery is a NL2SQL model pretrained on QA on free text that obviates the necessity of loading the entire dataset into memory and serializing databases.

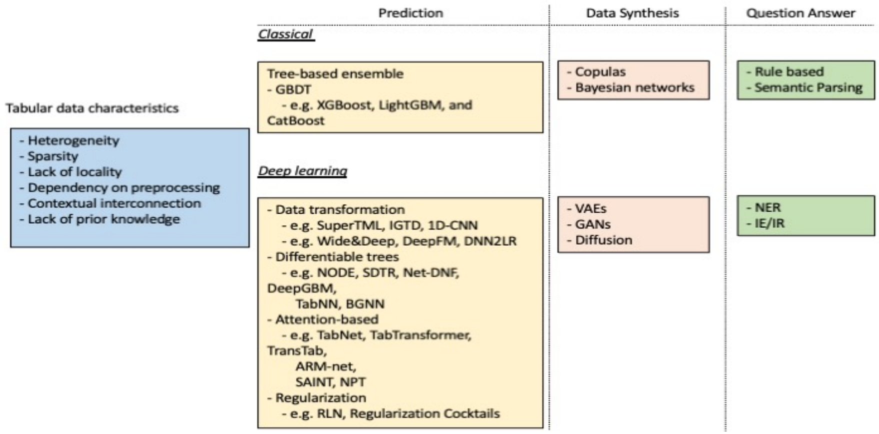

Figure 1: Tabular data characteristics and machine learning models for tabular data prediction, data synthesis and question answering before LLMs. JQ-TODO:please standardize the task naming in Figure1 and Figure2... e.g., QA vs. table understanding... just stick with one name is better

1.3 Overview of large language models (LLMs)

A language model (LM) is a probabilistic model that predicts the generative likelihood of future or missing tokens in a word sequence. Zhao et al. (2023b) thoroughly reviewed the development of LMs, and characterized the it into four different stages: The first stage is **Statistical Language Models (SLM)**, which learns the probability of word occurrence in an example sequence from previous words (e.g. N-Gram) based on Markov assumption (Saul & Pereira, 1997). Although a more accurate prediction can be achieved by increasing the context window, SML is limited by the curse of high dimensionality and high demand for computation power (Bengio et al., 2000). Next, **Neural Language Models (NLM)** utilize neural networks (e.g. Recurrent neural networks (RNN)) as a probabilistic classifier (Kim et al., 2016). In addition to learn the probabilistic function for word sequence, a key advantage of NLM is that they can learn the distributed representation (i.e. word embedding) of each word so that similar words are mapped close to each other in the embedding space (e.g. Word2Vec), making the model generalize well to unseen sequences that are not in the training data and help alleviate the curse of dimensionality (Bengio et al., 2000). Later, rather than learning a static word embedding, context-aware representation learning was introduced by pretraining the model on large-scale unannotated corpora using bidirectional LSTM that takes context into consideration (e.g., ELMo (Peters et al., 2018a)), which shows significant performance boost in various natural language processing (NLP) tasks (Wang et al., 2022a; Peters et al., 2018b). Along this line, several other **Pretrained Language Models (PLM)** were proposed utilizing a transformer architecture with self-attention mechanisms including BERT and GPT2 (Ding et al., 2023). The pre-training and fine-tuning paradigm, closely related to transfer learning, allows the model to gain general syntactic and semantic understanding of the text corpus and then be trained on task-specific objectives to adapt to various tasks. The final and most recent stage of LM is the **Large Language Models (LLMs)**, and will be the focus of this paper. Motivated by the observation that scaling the data and model size usually leads to improved performance, researchers sought to test the boundaries of PLM's performance of a larger size, such as text-to-text transfer transformers (T5) (Raffel et al., 2023), GPT-3 (Brown et al., 2020), etc. Intriguingly, some advanced abilities emerge as a result. These large-sized PLMs (i.e. LLMs) show unprecedentedly powerful capabilities (also called emergent abilities) that go beyond traditional language modeling and start to gain capability to solve more general and complex tasks which was not seen in PLM. Formally, we define a LLM as follows:

Definition 1 (Large Language Model). A large language model (LLM) M, parameterized by θ, is a Transformer-based model with an architecture that can be autoregressive, autoencoding, or encoder-decoder. It has been trained on a large corpus comprising hundreds of millions to trillions of tokens. LLMs encompass pre-trained models and for our survey, refers to models that have at least 1 billion parameters.

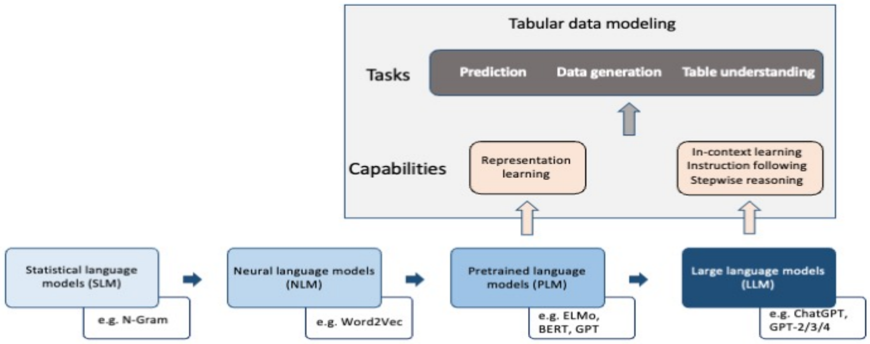

Figure 2: Development of language models and their applications in tabular data modeling.

Several key emergent abilities of LLMs are critical for data understanding and modeling including **in-context learning**, **instruction following**, and **multi-step reasoning**. In-context learning refers to designing large auto-regressive language models that generate responses on unseen task without gradient update, only learning through a natural language task description and a few in-context examples provided in the prompt. The GPT3 model (Brown et al., 2020) with 175 billion parameters presented an impressive in-context learning ability that was not seen in smaller models. LLMs have also demonstrated the ability to complete new tasks by following only the instructions of the task descriptions (also known as zero-shot prompts). Some papers also fine-tuned LLMs on a variety of tasks presented as instructions (Thoppilan et al., 2022). However, instruction-tuning is reported to work best only for larger-size models (Wei et al., 2022a; Chung et al., 2022). Solving complex tasks involving multiple steps have been challenging for LLMs. By including intermediate reasoning steps, prompting strategies such as chain-of-thought (CoT) has been shown to help unlock the LLM ability to tackle complex arithmetic, commonsense, and symbolic reasoning tasks (Wei et al., 2023). These new abilities of LLMs lay the groundwork for exploring their integration into intricate tasks extending beyond traditional NLP applications across diverse data types.

1.3.1 Applications of LLMs in tabular data

Despite the impressive capabilities of LM in addressing NLP tasks, its utilization for tabular data learning has been constrained by differences in the inherent data structure. Some research efforts have sought to utilize the generic semantic knowledge contained in PLM, predominantly BERT-based models, for modeling tabular data (Figure 2). This involves employing PLM to learn contextual representation with semantic information taking header information into account (Chen et al., 2020b). The typical approach includes transforming tabular data into text through serialization (detailed explanation in Section 2) and employing a masked-language-modeling (MLM) approach for fine-tuning the PLM, similar to that in BERT (PTab, CT-BERT, TABERT (Liu et al., 2022a; Ye et al., 2023a; Yin et al., 2020a). In addition to being able to incorporate semantic knowledge from column names, converting heterogenous tabular data into textual representation enables PLMs to accept inputs from diverse tables, thus enabling cross-table training. Also, due to the lack of locality property of tabular data, models need to exhibit permutation invariance of feature columns (Ye et al., 2023a). In this fashion, TABERT was proposed as a PLM trained on both natural language sentence and structured data (Yin et al., 2020a), PTab demonstrated the importance of cross-table training for an enhanced representation learning (Liu et al., 2022a), CT-BERT employs masked table modeling (MTM) and contrastive learning for cross-table pretraining that outperformed tree-based models (Ye et al., 2023a). However, previous research primarily focuses on using LM for representation learning, which is quite limited.

1.3.2 Opportunities for LLMs in tabular data modeling

Many studies today explore the potential of using LLMs for various tabular data tasks, ranging from prediction, data generation, to data understanding (further divided into question answering and data reasoning). This exploration is driven by LLMs' unique capabilities such as in-context learning, instruction following, and step-wise reasoning. The opportunities for applying LLMs to tabular data modeling are as follows:

1. Deep learning methods often exhibit suboptimal performance on datasets they were not initially trained on, making transfer learning using the pre-training and fine-tuning paradigm highly promising (Shwartz-Ziv & Armon, 2022).
2. The transformation of tabular data into LLM-readable natural language addresses the curse of dimensionality associated with one-hot encoding of high-dimensional categorical data during tabular preprocessing.
3. The emergent capabilities, such as step-by-step reasoning through CoT, have transformed LM from language modeling to a more general task-solving tool. Research is needed to test the limit of LLM's emergent abilities on tabular data modeling.

In the remainder of the article, we provide a comprehensive review of recent advancements in modeling tabular data using LLMs. In Section 2, we introduce key techniques related to the adaptation of tabular data for LLMs. Subsequently, we cover the applications of LLMs in prediction tasks (Section 3), data augmentation and enrichment tasks (Section 4), and question answering/table understanding tasks (Section 5). Finally, Section 6 discusses limitations and future directions, while Section 7 concludes.

1.4 Contribution

The key contributions of this work are as follows:

1. **A formal break down of key techniques for LLMs' applications on tabular data** We split the application of LLM in tabular data to tabular data prediction, tabular data synthesis, tabular data question answering and table understanding. We further extract key techniques that can apply to all applications. We organize these key techniques in a taxonomy that researchers and practitioners can leverage to describe their methods, find relevant techniques and understand the difference between these techniques. We further breakdown each technique to subsections so that researchers can easily find relevant benchmark techniques and properly categorize their proposed techniques.
2. **A survey and taxonomy of metrics for LLMs' applications on tabular data.** For each application, we categorize and discuss a wide range of metrics that can be used to evaluate the performance of that application. For each application, we documented the metric of all relevant methods, and we identify benefits/limitations of each class of metrics to capture application's performance. We also provide recommended metrics when necessary.
3. **A survey and taxonomy of datasets for LLMs' applications on tabular data.** For each application, we identify datasets that are commonly used for benchmark. For table understanding and question answering, we further categorize datasets by their downstream applications: Question Answering, Natural Language Generation, Classification, Natural Language Inference and Text2SQL. We further provided recommended datasets based on tasks and their GitHub link. Practitioners and researchers can look at the section and find relevant dataset easily.
4. **A survey and taxonomy of techniques for LLMs' applications on tabular data.** For each application, we break down an extensive range of tabular data modeling methods by steps. For example, tabular data prediction can be breakdown by pre-processing (modifying model inputs), target augmentation (modifying the outputs), fine-tuning (fine-tuning the model). We construct granular subcategories at each stage to draw similarities and trends between classes of methods,

<table>
<tr><th>Method</th><th>Description</th><th>Example</th><th>Papers that investigated this</th></tr>
<tr><td>DFLoader</td><td>Python code where a dictionary is loaded as a Pandas dataframe</td><td>pd.DataFrame({
name:[‘helen’], age:[47] })</td><td>Singha et al. (2023)</td></tr>
<tr><td>JSON</td><td>Row number as indexes, with each row represented as a dictionary of keys (column names) and values</td><td>{"0": {"name": "helen", "age": "47"}}</td><td>Singha et al. (2023); Sui et al. (2023b)</td></tr>
<tr><td>Data Matrix</td><td>Dataframe as a list of lists, where the firm item is the column header</td><td>[[‘’,‘name’,‘age’]
[0, ‘helen’, 47]]</td><td>Singha et al. (2023)</td></tr>
<tr><td>Markdown</td><td>Rows are line-separated, columns are separated by “|” [1]</td><td>|	name	age
0 |helen	47|</td><td>Singha et al. (2023); Liu et al. (2023e); Zhang et al. (2023d); Ye et al. (2023b); Zhao et al. (2023d); Sui et al. (2023b)</td></tr>
<tr><td>X-Separated</td><td>Rows are line-separated, columns are separated by “,”, “\t”, “:”, etc.</td><td>, name, age
0, helen, 47</td><td>Singha et al. (2023); Narayan et al. (2022)</td></tr>
<tr><td>Attribute-Value Pairs</td><td>Concatenation of paired columns and cells {c : v}</td><td>name:helen ; age:47</td><td>Wang et al. (2023c)</td></tr>
<tr><td>HTML</td><td>HTML element for tabular data</td><td><table><thead><tr><th></th>
<th>name</th><th>age</th></tr>
</thead><tbody><tr><th>0</th>
<td>helen</td><td>47</td></tr>
</tbody></table></td><td>Singha et al. (2023); Sui et al. (2023c;b)</td></tr>
<tr><td>Sentences</td><td>Rows are converted into sentences using templates</td><td>name is helen, age is 47</td><td>Yu et al. (2023); Hegselmann et al. (2023); Gong et al. (2020)</td></tr>
</table>

Table 1: Text-based serialization methods.

and with illustrated examples of main techniques. Practitioners and researchers can look at the section and understand the difference of each technique. We only recommend benchmark methods and provide GitHub link of these techniques for reference and benchmark.

5. **An overview of key open problems and challenges that future work should address.** We challenge future research to solve bias problem in tabular data modeling, mitigate hallucination, find better representations of numerical data, improve capacity, form standard benchmark, improve model interpretability, create an integrated workflow, design better fine-tuning strategies and improve the performance of downstream applications.

2 Key techniques for LLMs' applications on tabular data

While conducting our survey, we noticed a few common components in modeling tabular data with LLMs across tasks. We discuss common techniques, like serialization, table manipulations, prompt engineering, and building end-to-end systems in this section. Fine-tuning LLMs is also popular, but tend to be application-specific, so we leave discussions about it to Sections 3 and 5.

2.1 Serialization

Since LLMs are sequence-to-sequence models, in order to feed tabular data as inputs into an LLM, we have to convert the structured tabular data into a text format (Sui et al., 2023b; Jaitly et al., 2023).

Text-based Table 1 describes the common text-based serialization methods in the literature. A straightforward way would be to directly input a programming language readable data structure (E.g. Pandas DataFrame Loader for Python, line-separated JSON-file format, Data Matrix represented by a list of lists, HTML code reflecting tables, etc). Alternatively, the table could be converted into X-separated values, where X could be any reasonable delimiter like comma or tab. Some papers convert the tables into human-readable sentences using templates based on the column headers and cell values. The most common approach based on our survey is the Markdown format.

Embedding-based Many papers also employ table encoders, which were fine-tuned from PLMs, to encode tabular data into numerical representations as the input for LLMs. There are multiple table encoders, built on BERT (Devlin et al., 2019) for table-related task, like TAPAS (Herzig et al., 2020), TABERT (Yin et al., 2020b), TURL (Deng et al., 2022a), TUTA (Wang et al., 2021), TABBIE (Iida et al., 2021) and UTP (Chen et al., 2023a). For LLMs with >1B parameters, there are UniTabPT (Sarkar & Lausen, 2023) with 3B parameters (based on T5 and Flan-T5 models)), TableGPT (Gong et al., 2020) with 1.5B parameters (based on GPT2), and TableGPT[2] (Zha et al., 2023) with 7B parameters (based on Phoenix (Chen et al., 2023b)).

Graph-based & Tree-based A possible, but less commonly explored, serialization method involves converting a table to a graph or tree data structure. However, when working with sequence-to-sequence models, these structures must still be converted back to text. For Zhao et al. (2023a), after converting the table into a tree, each cell's hierarchical structure, position information, and content was represented as a tuple and fed into GPT3.5.

Comparisons Research has shown that LLM performance is sensitive to the input tabular formats. Singha et al. (2023) found that DFLoader and JSON formats are better for fact-finding and table transformation tasks. Meanwhile, Sui et al. (2023a) found that HTML or XML table formats are better understood by GPT models over tabular QA and FV tasks. However, they require increased token consumption. Likewise, Sui et al. (2023b) also found markup languages, specifically HTML, outperformed X-separated formats for GPT3.5 and GPT4. Their hypothesis is that the GPT models were trained on a significant amount of web data and thus, probably exposed the LLMs to more HTML and XML formats when interpreting tables.

Apart from manual templates, Hegselmann et al. (2023) also used LLMs (Fine-tuned BLOOM on ToTTo (Parikh et al., 2020b), T0++ (Sanh et al., 2022), GPT-3 (Ouyang et al., 2022)) to generate descriptions of a table as sentences, blurring the line between a text-based and embedding-based serialization methodology. However, for the few-shot classification task, they find that traditional list and text templates outperformed the LLM-based serialization method. Amongst LLMs, the more complex and larger the LLM, the better the performance (GPT-3 has 175B, T0 11B, and fine-tuned BLOOM model 0.56B parameters). A key reason why the LLMs are worse off at serializing tables to sentences is due to the tendency for LLMs to hallucinate: LLMs respond with unrelated expressions, adding new data, or return unfaithful features.

2.2 Table Manipulations

One important characteristic of tabular data is its heterogeneity in structure and content. They oftentimes come in large size with different dimensions encompassing various feature types. In order for LLMs to ingest tabular data efficiently, it is important to compact tables to fit context lengths, for better performance and reduced costs.

Compacting tables to fit context lengths, for better performance and reduced costs For smaller tables, it might be possible to include the whole table within a prompt. However, for larger tables, there are three challenges:

Firstly, some models have short context lengths (E.g. Flan-UL2 (Tay et al., 2023b) supports 2048 tokens, Llama 2 (Touvron et al., 2023b) supports 4096 context tokens) and even models that support large context lengths might still be insufficient if the table is over say 200K rows (Claude 2.1 supports up to 200K tokens).

Secondly, even if the table could fit the context length, most LLMs are inefficient in dealing with long sentences due to the quadratic complexity of self-attention (Sui et al., 2023b; Tay et al., 2023a; Vaswani et al., 2017). When dealing with long contexts, performance of LLMs significantly degrades when models must access relevant information in the middle of long contexts, even for explicitly long-context models (Liu et al., 2023b). For tabular data, Cheng et al. (2023); Sui et al. (2023c) highlights that noisy information becomes an issue in large tables for LMs. Chen (2023) found that for table sizes beyond 1000 tokens, GPT-3's performance degrades to random guesses.

Thirdly, longer prompts incur higher costs, especially for applications built upon LLM APIs.

[2]Same name, different group of authors.

To address these issues, Herzig et al. (2020); Liu et al. (2022c) proposed naive methods to truncate the input based on a maximum sequence length. Sui et al. (2023b) introduced predefined certain constraints to meet the LLM call request. Another strategy is to do search and retrieval of only highly relevant tables, rows, columns or cells which we will discuss later in Section 5.

Additional information about tables for better performance Apart from the table, some papers explored including table schemas and statistics as part of the prompt. Sui et al. (2023c) explored including additional information about the tables: Information like " dimension, measure, semantic field type" help the LLM achieve higher accuracy across all six datasets explored. "statistics features" improved performance for tasks and datasets that include a higher proportion of statistical cell contents, like FEVEROUS (Aly et al., 2021). Meanwhile, "document references" and "term explanations" add context and semantic meaning to the tables. "Table size" had minimal improvements, while "header hierarchy" added unnecessary complexity, and hurt performance.

Robustness of LLM performance to table manipulations Liu et al. (2023e) critically analyzed the robustness of GPT3.5 across structural perturbations in tables (transpose and shuffle). They find that LLMs suffer from structural bias in the interpretation of table orientations, and when tasked to transpose the table, LLMs performs miserably (50% accuracy). However, LLMs can identify if the first row or first column is the header (94-97% accuracy). Zhao et al. (2023e) investigated the effects of SOTA Table QA models on manipulations on the table header, table content and natural language question (phrasing).[3] They find that all examined Table QA models (TaPas, TableFormer, TaPEX, OmniTab, GPT3) are not robust under adversarial attacks.

2.3 Prompt Engineering

A prompt is an input text that is fed into an LLM. Designing an effective prompt is a non-trivial task, and many research topics have branched out from prompt engineering alone. In this subsection, we cover the popular techniques in prompt engineering, and how researchers have used them for tasks involving tables.

Prompt format The simplest format is concatenating task description with the serialized table as string. An LLM would then attempt to perform the task described and return a text-based answer. Clearly-defined and well-formatted task descriptions are reported to be effective prompts (Marvin et al., 2023). Some other strategies to improve performance are described in the next few paragraphs. Sui et al. (2023b) recommended that external information (such as questions and statements) should be placed before the tables in prompts for better performance.

In-context learning As one of the emergent abilities of LLMs (see 1.3), in-context learning refers to incorporate similar examples to help the LLMs understand the desired output. Sui et al. (2023b) observed significant performance drops performance, of overall accuracy decrease of 30.38% on all tasks, when changing their prompts from a 1-shot to a 0-shot setting. In terms of choosing appropriate examples, Narayan et al. (2022) found their manually curated examples to outperform randomly selected examples by an average of 14.7 F1 points. For Chen (2023), increasing from 1-shot to 2-shot can often benefit the model, however, further increases did not lead to more performance gain.

Chain-of-Thought and Self-consistency Chain-of-Thought (CoT) (Wei et al., 2022c) induces LLMs to decompose a task by performing step-by-step thinking, resulting in better reasoning. Program-of-Thoughts (Chen et al., 2022) guides the LLMs using code-related comments like "*Let's write a program step-by-step...*". Zhao et al. (2023d) explored CoT and PoT strategies for the numerical QA task. Yang et al. (2023) prompt the LLMs with one shot CoT demonstration example to generate a reasoning and answer. Subsequently,

[3]For table headers, they explored synonym and abbreviation replacement perturbations. For table content, they explored five perturbations: (1) row shuffling, (2) column shuffling, (3) extending column names content into semantically equivalent expressions, (4) masking correlated columns (E.g. "Ranking" and "Total Points" can be inferred from one another), and (5) introducing new columns that are derived from existing columns. For the question itself, they perturbed questions at the word-level or sentence-level.

they included the reasoning texts, indicated by special "<CoT>" token, as part of inputs to fine-tune smaller models to generate the final answer.

Self-consistency (SC) (Wang et al., 2023b) leverages the intuition that a complex reasoning problem typically admits multiple different ways of thinking leading to its unique correct answer. SC samples a diverse set of reasoning paths from an LLM, then selects the most consistent answer by marginalizing out the sampled reasoning paths. Inspired by these strategies, Zhao et al. (2023a); Ye et al. (2023b) experimented with multi-turn dialogue strategies, where they decompose the original question into sub-tasks or sub-questions to guide the LLM's reasoning. Sui et al. (2023c) instructed the LLM to "*identify critical values and ranges of the last table related to the statement*" to obtain additional information that were fed to the final LLM, obtaining increased scores for five datasets. Liu et al. (2023e) also investigated strategies around SC, along with self-evaluation, which guides the LLM to choose between the two reasoning approaches based on the question's nature and each answer's clarity. Deng et al. (2022b) did consensus voting across a sample a set of candidate sequences, then selected final response by ensembling the derived response based on plurality voting.

Chen (2023) investigated the effects of both CoT and SC on QA and FV tasks. When investigating the explainability of LLM's predictions, Dinh et al. (2022) experimented with a multi-turn approach of asking GPT3 to explain its own prediction from the previous round, and guided the explanation response using CoT by adding the line "*Let's think logically. This is because*".

Retrieval-augmented generation (RAG) Retrieval-augmented generation (RAG) relies on the intuition that the LLMs are general models, but can be guided to a domain-specific answer if the user includes the relevant context within the prompts. By incorporating tables as part of the prompts, most papers described in this survey can be attributed as RAG systems. A particular trait challenge in RAG is to extract the most relevant information out of a large pool of data to better inform the LLMs. This challenge overlaps slightly with the strategies about table sampling mentioned earlier under Section 2.2. Apart from the aforementioned methods, Sundar & Heck (2023) designed a dual-encoder-based Dense Table Retrieval (DTR) model to rank cells of the table to the relevance of the query. The ranked knowledge sources are incorporated within the prompt, and led to top ROUGE scores.

Role-play Another popular prompt engineering technique is role-play, which refers to including descriptions in the prompt about the person the LLM should portray as it completes a task. For example, Zhao et al. (2023a) experimented with the prompt "*Suppose you are an expert in statistical analysis.*".

2.4 End-to-end systems

Since LLMs can generate any text-based output, apart from generating human-readable responses, it could also generate code readable by other programs. Abraham et al. (2022) designed a model that converts natural language queries to structured queries, which can be run against a database or a spreadsheet. Liu et al. (2023e) designed a system where the LLM could interact with Python to execute commands, process data, and scrutinize results (within a Pandas DataFrame), iteratively over a maximum of five iterations. Zhang et al. (2023d) demonstrated that we can obtain errors from the SQL tool to be fed back to the LLMs. By implementing this iterative process of calling LLMs, they improved the success rate of the SQL query generation. Finally, Liu et al. (2023c) proposes a no-code data analytics platform that uses LLMs to generate data summaries, including generating pertinent questions required for analysis, and queries into the data parser. A survey by Zhang et al. (2023g) covers further concepts about natural language interfaces for tabular data querying and visualization, diving deeper into recent advancements in Text-to-SQL and Text-to-Vis domains.

3 LLMs for predictions

Several studies endeavor to leverage LLMs for prediction task from tabular data. This section will delve into the existing methodologies and advancements pertaining to two categories of tabular data: standard feature-based tabular data and time series data. Time series prediction is different from normal feature-based tabular data since the predictive power heavily rely on pastime series numbers. For each category, we divide it to different steps which includes preprocessing, fine-tuning and target augmentation. Preprocessing explains how different prediction methods generate input to the language model. Preprocessing includes serialization, table manipulation and prompt engineering. Target augmentation maps the textual output from LLMs to a target label for prediction tasks. At the end, we will briefly touch on domain specific prediction methods using LLMs.

3.1 Dataset

For task specific fine-tuning, most datasets for prediction task are chosen from UCI ML, OpenML or a combo of 9 datasets created by Manikandan et al. (2023). We put all details in Table 2. Using the combo of 9 datasets is recommended [4] since it contains larger size dataset and more diverse feature set compared to OpenML and UCI ML. For general finetuning, existed methods choose Kaggle API[5] as it has 169 datasets and Datasets are very diverse.

Dataset	Dataset Number	Papers that used this dataset
OpenML	11	Dinh et al. (2022); Manikandan et al. (2023)
Kaggle API	169	Hegselmann et al. (2023); Wang et al. (2023a); Zhang et al. (2023a)
Combo	9	Hegselmann et al. (2023); Wang et al. (2023a); Zhang et al. (2023a)
UCI ML	20	Manikandan et al. (2023); Slack & Singh (2023)
DDX	10	Slack & Singh (2023)

Table 2: Combo is the combination of the following dataset in the form of dataset name (number of rows, number of features): Bank (45,211 rows, 16 feats), Blood (748, 4), California (20,640, 8), Car (1,728, 8), Creditg (1,000, 20), Income (48,842, 14), and Jungle (44,819, 6), Diabetes (768, 8) and Heart (918, 11).

3.2 Tabular prediction

Algorithm	Type	Method	Resource	Metric	Used Model
TabletSlack & Singh (2023)	Tabular	No Finetune	Low	F1	GPTJ/Tk-Instruct/Flan T5
SummaryBoostManikandan et al. (2023)	Tabular	No Finetune	High	RMSE	GPT3
LIFTDinh et al. (2022)	Tabular	Finetune	High	MAE/RMSE	GPT3/GPTJ
TabLLMHegselmann et al. (2023)	Tabular	Finetune	High	AUC	GPT3/T0
UnipredictWang et al. (2023a)	Tabular	Finetune	Low	ACC	GPT2
GTLZhang et al. (2023a)	Tabular	Finetune	Low	ACC	LLaMA
SerializeLLMJaitly et al. (2023)	Tabular	Finetune	High	AUC	T0
PromptCastXue & Salim (2022)	Time Series	Finetune	High	MAE/ RMSE/ Missing Rate	T5/Bigbird/LED
ZeroTSGruver et al. (2023)	Time Series	No Finetune	Low	MAE/ Scale MAE/ CRPS	GPT3/LLAMA2
TESTSun et al. (2023a)	Time Series	Finetune	High	ACC/ RMSE	Bert/ GPT2/ ChatGLM/ LLaMa
TimeLLMJin et al. (2023a)	Time Series	Finetune	High	SMAPE/ MSAE/ OWA	LLAMA7B/ GPT2
MediTabWang et al. (2023c)	Medical	Finetune	High	PRAUC/AUCROC	BioBert/GPT3.5/UnifiedQA-v2-T5
CTRLLi et al. (2023)	Finance	Finetune	High	AUC/LogLoss	Roberta/ChatGLM
FinPTYin et al. (2023)	CTR	Finetune	High	F1 Score	FlanT5/ChatGPT/GPT4

Table 3: Prediction methods. Resource is high if it has to finetune a model with size $\geq$ 1B even if it is PEFT. Used Model include all models used in the paper which includes serialization, preprocessing and model finetuning. ACC stands for accuracy. AUC stands for Area under the ROC Curve. MAE stands for mean absolute error. RMSE stands for root-mean-square error. F1 score is calculated from the precision and recall of the test, where the precision is the number of true positive results divided by the number of all samples predicted to be positive, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive. CRPS is continous ranked probability score. We will introduce other metrics in relevant sections.

Preprocessing Serialization in prediction task is mostly Text-based in section 2.1. Table manipulation includes statistics and metadata of datasets in section 2.2. Prompt engineering includes task specific cues and relevant samples in section 2.1. We give an illustration of different preprocessing methods in Table 4

[4]Here is the GitHub repository to get the data `https://Github.com/clinicalml/TabLLM/tree/main/datasets`
[5]Here is the website to get the pretrained data `https://Github.com/Kaggle/kaggle-api`

As one of the earliest endeavors, LIFT (Dinh et al., 2022) tried a few different serialization methods, such as feature and value as a natural sentence such as "The column name is Value" or a bunch of equations, such as $col_1 = val_1, col_2 = val_2, ...$ The former is shown to achieve higher prediction accuracy, especially in low-dimensional tasks. The same conclusion was also found by TabLLM (Hegselmann et al., 2023) where they evaluated 9 different serialization methods. They found that a textual enumeration of all features - 'The column name is Value', performs the best. They also added a description for classification problem. For medical prediction, they mimic the thinking process of medical professional as prompt engineering. They found out that LLM actually make use for column name and their relationships in few shot learning settings. In a subsequent study, TABLET (Slack & Singh, 2023) included naturally occurring instructions along with examples for serialization. In this case, where the task is for medical condition prediction, naturally occurring instructions are from consumer-friendly sources, such as government health website or technical reference such as Merck Manual. It includes instructions, examples, and test data point. They found that these instructions significantly enhance zero-shot F1 performance. However, LLMs still ignore instructions sometimes, leading to prediction failures. Along this fashion, more studies tested a more complex serialization and prompt engineering method rather than simple concatenation of feature and value for serialization. The schema-based prompt engineering usually includes background information of the dataset, a task description, a summary, and example data points. Summary Boosting(Manikandan et al., 2023) serializes data and metadata into text prompts for summary generation. This includes categorizing numerical features and using a representative dataset subset selected via weighted stratified sampling based on language embeddings. Serilize-LM (Jaitly et al., 2023) introduces 3 novel serialization techniques which boosts LLM performance in domain specific datasets. They included related features into one sentence to make the prompt more descriptive and easier to understand for LLM. Take car classification as an example, attributes like make, color and body type are now combined into a single richer sentence. It leverages covariance to identify most relevant features and either label them critical or adding a sentence to explain the most important features. Finally, they converted tabular data into LaTeX code format. This LaTeX representation of the table was then used as the input for fine-tuning our LLM by just passing a row representation preceded by `hline` tag without any headers. UniPredict (Wang et al., 2023a) reformats meta data by consolidating arbitrary input M to a description of the target and the semantic descriptions of features. Feature serialization follows a "column name is value" format, ... The objective is to minimize the difference between the output sequence generated by the adapted LLM function and the reference output sequence generated from target augmentation (represented by serialize target). Generative Tabular Learning (GTL) was proposed by (Zhang et al., 2023a) which includes two parts: 1) the first part specifies the task background and description with optionally some examples as in-context examples(Prompt Engineering); 2) the second part describes feature meanings and values of the current instance to be inferred(Serialization); For researchers and practitioners, we recommend to benchmark LIFT, TABLET and TabLLM for new preprocessing method since their methods are representative and clearly documented. The code is available. [6]

Some other methods leverage an LLM to rewrite the serialization or do the prompt engineering. TabLLM (Hegselmann et al., 2023) showed that LLM is not good for serialization because it is not faithful and may hallucinate. Summary Boosting(Manikandan et al., 2023) uses GPT3 to convert metadata to data description and generate summary for a subset of datasets in each sample round. TABLET (Slack & Singh, 2023) fits a simple model such as one layer rule set morel or prototype with 10 most important features on the task's full training data. It then serializes the logic into text using a template and revise the templates using GPT3. Based on their experiments, generated instructions do not significantly improve the performance. Thus, unless the serialization requires summarizing the long input, it is not recommended to use LLM to rewrite serialization.

Target Augmentation LLMs can solve complex task through text generation, however, the output is not always controllable (Dinh et al., 2022). As a result, mapping the textual output from LLMs to a target label for prediction tasks is essential. We call it target augmentation. A straightforward but labor-intensive way is manual labeling as used by Serilize-LM (Jaitly et al., 2023). LIFT (Dinh et al., 2022) employs ### and @@@ for question-answer separation and end of generation, respectively, placing answers in between. To mitigate invalid inferences, LIFT conducts five inference attempts, defaulting to the training set's average

[6] Here is the Github repo for TABLET `https://Github.com/dylan-slack/Tablet`, TabLLM `https://Github.com/clinicalml/TabLLM` and LIFT `https://Github.com/UW-Madison-Lee-Lab/LanguageInterfacedFineTuning`

value if all fail. TabLLM (Hegselmann et al., 2023) uses verbalizer (Cui et al., 2022) to map the answer to a valid class. UniPredict (Wang et al., 2023a) has the most complicated target augmentation. They transform the target label into a set of probabilities for each class via a function called "augment". Formally, for target T in an arbitrary dataset D, they define a function $augment(T) = C, P$, where C are new categories of targets with semantic meaning and P are the assigned probabilities to each category. They extend the target into categorical one-hot encoding and then use an external predictor to create the calibrated probability distributions. This replaces the 0/1 one-hot encoding while maintaining the final prediction outcome. Formally, given the target classes $t \in 0, ..., |C|$ and target probabilities $p \in P$, they define a function serialize target(t, p) that serializes target classes and probabilities into a sequence formatted as "class $t_1 : p_1$, $t_2 : p_2$, . . . " We give an example for each method in 5 While customized target augmentation could be useful in some cases, the simple Verbalizer is recommended for its convenience to implement and can assign the probability of the output.

Inference Only Prediction Some work uses LLMs directly for prediction without fine-tuning, we refer these approaches inference only prediction. TABLET (Slack & Singh, 2023) utilizes models like Tk-Instruct (Wang et al., 2022b) 11b, Flan-T5 (Chung et al., 2022) 11b, GPT-J (Black et al., 2022) 6b, and ChatGPT to inference the model, but find out that a KNN approach with feature weights from XGBoost surpasses Flan-T5 11b in performance using similar examples and instructions. Summary Boosting (Manikandan et al., 2023) creates multiple input through serialization step. The AdaBoost algorithm then creates an ensemble of summary-based weak learners. While non-fine-tuned LLMs struggle with continuous attributes, summary boosting is effective with smaller datasets. Furthermore, its performance is enhanced using GPT-generated descriptions by leveraging existing model knowledge, underscoring the potential of LLMs in new domains with limited data. However, it does not perform well when there are many continuous variables. For any new LLM based prediction method without any fine-tuning, we suggest to benchmark LIFT and TABLET. LIFT is the first LLM based method for inference only prediction. TABLET shows significantly better performance compared to LIFT. Both methods have code available.

Fine-tuning For studies involving fine-tuning, they typically employ one of two distinct approaches. The first involves training a LLM model on large datasets to learn fundamental features before adapting it to specific prediction tasks. The second takes a pre-trained LLM and further training it on a smaller, specific prediction dataset to specialize its knowledge and improve its performance on the prediction. LIFT (Dinh et al., 2022) fine-tunes pretrained language models like GPT-3 and GPT-J using Low-Rank Adaptation (LoRA) on training set. They found that LLM with general pretraining could improve the performance. However, the performance of this method does not surpass in context learning result. TabLLM (Hegselmann et al., 2023) uses T0 model (Sanh et al., 2021) and t few (**?**) for fine-tuning. TabLLM has demonstrated remarkable few-shot learning capabilities outperforming traditional deep-learning methods and gradient-boosted trees. TabLLM's efficacy is highlighted by its ability to leverage the extensive knowledge encoded in pre-trained LLMs, requiring minimal labeled data. However, the sample efficiency of TabLLM is highly task-dependent. Jaitly et al. (2023) uses T0 (Sanh et al., 2021). It is trained using Intrinsic Attention-based Prompt Tuning (IA3) (Liu et al., 2022b). However, this method only works for few short learning, worse than baseline when number of shots more or equal to 128. T0 model (Sanh et al., 2021) is commonly used as base model for tabular prediction fine-tuning.

UniPredict (Wang et al., 2023a) trains a single LLM (GPT2) on an aggregation of 169 tabular datasets with diverse targets and observe advantage over existed methods. This model does not require fine-tuning LLM on specific datasets. Model accuracy and ranking is better than XGBoost when the number of samples is small. The model with target augmentation performs noticeably better than the model without augmentation. It does not perform well when there are too many columns or fewer representative features. TabFMs (Zhang et al., 2023a) fine-tunes LLaMA to predict next token. we are left with 115 tabular datasets. To balance the number of instances across different datasets, we randomly sample up to 2,048 instances from each tabular dataset for GTL. They employed GTL which significantly improves LLaMA in most zero-shot scenarios. Based on the current evidence, we believe that fine-tuning on large number of datasets could further improve the performance. However, both UniPredict and GTL have not released their code yet.

Metric We suggest to report AUC for classification prediction and RMSE for regression since they are mostly common used in the literature 3

Methodology	Method	Example
Feature name + Feature Value + Predicted Feature Name	Dinh et al. (2022); Hegselmann et al. (2023)	Car Brand is Land Rover. Year is 2017. Repair claim is
Task Background + Feature meaning + Feature Value + Predicted Feature meaning	Zhang et al. (2023a)	The task is about fraud repair claim prediction. The brand of car is Land Rover. The produce year is 2017. The repair claim of the car is
Dataset Summary + LLM Processed Feature + Task	Manikandan et al. (2023)	Larger car is always more expensive. This is a 2017 Land Rover. Therefore, this car repair claim is (Fraudulent or Not Fraudulent):
Latex Format of features value + Task	Jaitly et al. (2023)	\hline Land Rover & 2017 ... Is this car repair claim fraudulent? Yes or No?
Expert Task Understanding + Label + Task	Slack & Singh (2023)	Identify if car repair claim is fraudulent. Older car is more likely to have fraudulent repair claim. Features Car Brand: Land Rover Year: 2017. Answer with one of the following: Yes | No
Dataset description + Feature meaning + Feature Value + Task	Wang et al. (2023a)	The dataset is about fraud repair claim. Car Brand is the brand of car. Year is the age when the car is produced. The features are: Car Brand is Land Rover. Year is 2017. Predict if this car repair claim fraudulent by Yes for fraudulent, No for not fraudulent

Table 4: Method and Example for different preprocessing in general prediction. The example is to predict if a car repair claim fraudulent or not.

3.3 Time Series Forecasting

Compared to prediction on feature-based tabular data with numerical and categorical features, time series prediction pays more attention to numerical features and temporal relations. Thus, serialization and target augmentation are more relevant to how to best represent numerical features. Many papers have claimed that they use LLM for time series. However, most of these papers use LLM that is smaller than 1B. We will not discuss these methods here. Please refer to (Jin et al., 2023b) for a complete introduction of these methods.

Preprocessing PromptCast (Xue & Salim, 2022) uses input time series data as it is and convert it to a test format with minimal description of the task and convert target as a sentence to be the output. ZeroTS (Gruver et al., 2023) claims that the number is not encoded well in original LLM encoding method. Thus, it encodes numbers by breaking them down by a few digits or by each single digit for GPT-3 and LLaMA, respectively. It uses spaces and commas for separation and omitting decimal points. Time LLM (Jin et al., 2023a) involves patching time series into embeddings and integrating them with word embeddings to create a comprehensive input. This input is complemented by dataset context, task instructions, and input statistics as a prefix. TEST (Sun et al., 2023a) introduces an embedding layer tailored for LLMs, using exponentially dilated causal convolution networks for time series processing. The embedding is generated through contrastive learning with unique positive pairs and aligning text and time series tokens using similarity measures. Serialization involves two QA templates, treating multivariate time series as univariate series for sequential template filling.

Target Augmentation In terms of output mapping, ZeroTS (Gruver et al., 2023) involves drawing multiple samples and using statistical methods or quantiles for point estimates or ranges. For Time-LLM (Jin et al., 2023a), the output processing is done through flatten and linear projection. The target augmentation method of ZeroTS is easy to implement [7] while TimeLLM's code is not available.

Inference Only Prediction Similar to feature-based tabular prediction, researchers explored LLMs' performance for time series forecasting without fine-tuning. ZeroTS (Gruver et al., 2023) examines the use of LLMs like GPT-3 (Brown et al., 2020) and LLaMA-70B Touvron et al. (2023a) directly for time series forecasting. It evaluates models using mean absolute error (MAE), Scale MAE, and continuous ranked probability score (CRPS), noting LLMs' preference for simple rule-based completions and their tendency towards repetition and capturing trends. The study notes LLMs' ability to capture time series data distributions and handle missing data without special treatment. However, this approach is constrained by window size and arithmetic ability, preventing it from further improvement.

Fine-tuning Fine-tuning the model for time series prediction is more commonly seen in current research. PromptCast (Xue & Salim, 2022) tried the method on inference only prediction or fine-tuning on task specific datasets. It shows that larger model always perform better. Time LLM (Jin et al., 2023a) presents a novel approach to time series forecasting by fine-tuning LLMs like LLaMa Touvron et al. (2023a) and GPT-2 (Brown et al., 2020). Time-LLM is evaluated using metrics symmetric mean absolute percentage error (SMAPE), mean absolute scaled error (MSAE), and overall weighted average (OWA). It demonstrates notable performance in few-shot learning scenarios, where only 5 percent or 10 percent of the data are used. This innovative technique underscores the versatility of LLMs in handling complex forecasting tasks. For TEST (Sun et al., 2023a), soft prompts are used for fine-tuning. The paper evaluates models like Bert, GPT-2 (Brown et al., 2020), ChatGLM (Zeng et al., 2023), and LLaMa Touvron et al. (2023a), using metrics like classification accuracy and RMSE. However, the result shows that this method is not as efficient and accurate as training a small task-oriented model. In general, currently LLaMa is the most commonly used model and soft prompt seems to be a suitable approach for fine-tuning.

Metric MAE is the most common metric. Continuous Ranked Probability Score (CRPS) as it captures distributional qualities, allowing for comparison of models that generate samples without likelihoods. CRPS is considered an improvement over MAE as it does not ignore the structures in data like correlations between time steps. Symmetric Mean Absolute Percentage Error (SMAPE) measures the accuracy based on percentage errors, Mean Absolute Scaled Error (MASE) is a scale-independent error metric normalized by the in-sample mean absolute error of a naive benchmark model, and Overall Weighted Average (OWA) is a combined metric that averages the ranks of SMAPE and MASE to compare the performance of different methods. Despite the introduction of new metrics, MAE and RMSE are mostly common used in the literature. We still recommend using MAE and RMSE as they are simple to implement and easy to benchmark.

Method	**Used Paper**	**Example**
Adding Special Token before and after the answer	Dinh et al. (2022)	`### {Category} @@@`
Verbalizer	Hegselmann et al. (2023)	`Output -> {category1: probability1, .}`
Specific Prefix	Manikandan et al. (2023); Slack & Singh (2023)	Please answer with category 1, category 2, ...
Predict probability and recalibrate	Wang et al. (2023a)	`{category1: probability1}` => Calibrated by XGBoost

Table 5: Target Augmentation method, used papers and examples

3.4 Application of Prediction using LLM

Medical Prediction It was found that PTL such as DeBERTa has been shown perform better than XGBoost in electronic health record (EHR) prediction tasks (McMaster et al., 2023). For preprocessing, Meditab Wang et al. (2023c) utilizes GPT-3.5 Brown et al. (2020) to convert tabular data into textual format, with a focus on extracting key values. Subsequently, it employs techniques such as linearization, prompting, and sanity checks to ensure accuracy and mitigate errors. For fine-tuning, the system further leverages multitask

[7] The code is in `https://Github.com/ngruver/llmtime`

learning on domain-specific datasets, generates pseudo-labels for additional data, and refines them using data Shapley scores. Pretraining on the refined dataset is followed by fine-tuning using the original data. The resulting model supports both zero-shot and few-shot learning for new datasets. GPT-3.5 accessed via OpenAI's API facilitates data consolidation and augmentation, while UnifiedQA-v2-T5 Khashabi et al. (2022) is employed for sanity checks. Additionally, Meditab utilizes a pretrained BioBert classifier Lee et al. (2019). The system undergoes thorough evaluation across supervised, few-shot, and zero-shot learning scenarios within the medical domain, demonstrating superior performance compared to gradient boosting methods and existing LLM-based approaches. However, it may have limited applicability beyond the medical domain. We recommend exploring the provided code[8] for tabular prediction tasks specifically in the medical domain. On top AUCROC, they also use precision recall curve (PRAUC) for evaluation. PRAUC is useful in imbalanced datasets which are always the case for medical data.

Financial Prediction FinPT (Yin et al., 2023) presents an LLM based approach to financial risk prediction. The method involves filling tabular financial data into a pre-defined template, prompting LLMs like ChatGPT and GPT-4 to generate natural-language customer profiles. These profiles are then used to fine-tune large foundation models such as BERT (Devlin et al., 2019), employing the models' official tokenizers. The process enhances the ability of these models to predict financial risks, with Flan-T5 emerging as the most effective backbone model in this context, particularly across eight datasets. For financial data, we suggest to use [9] and benchmark against FinPT[10].

Recommendation Prediction CTRL (Li et al., 2023) proposes a novel method for Click Through Rate (CTR) prediction by converting tabular data into text using human-designed prompts, making it understandable for language models. The model treats tabular data and generated textual data as separate modalities, feeding them into a collaborative CTR model and a pre-trained language model such as ChatGLM (Zeng et al., 2023), respectively. CTRL employs a two-stage training process: the first stage involves cross-modal contrastive learning for fine-grained knowledge alignment, while the second stage focuses on fine-tuning a lightweight collaborative model for downstream tasks. The approach outperforms all the SOTA baselines including semantic and collaborative models over three datasets by a significant margin, showing superior prediction capabilities and proving the effectiveness of the paradigm of combining collaborative and semantic signals. However, the code for this method is not available. They use LogLoss and AUC to evaluate the method. For LogLoss, A lower bound of 0 for Logloss indicates that the two distributions are perfectly matched, and a smaller value indicates a better performance.

4 LLMs for tabular data synthesis

In this section, we focus on the pivotal role of data synthesis. The escalating demand for nuanced datasets prompts the exploration of novel methodologies leveraging LLMs to augment tabular data. This section scrutinizes methodologies illuminating the transformative potential of conjoining LLMs and tabular data for data synthesis.

	Used LLM	Fine-tuned or not	Serialization	Metric
GReaT (Borisov et al., 2023b)	GPT2/DistilGPT2	Fine-tuned	Sentences	DCR, MLE
REaLTabFormer (Solatorio & Dupriez, 2023)	GPT2	Fine-tuned		DCR, MLE
TAPTAP (Zhang et al., 2023e)	GPT2/DistilGPT2	Fine-tuned	Sentences	DCR, MLE
TabuLa (Zhao et al., 2023f)	DistilGPT2	Fine-tuned	X-Separated	MLE
CLLM (Seedat et al., 2023)	GPT4	Non Fine-tuned	X-Separated	MLE
TabMT (Gulati & Roysdon, 2023)	Masked Transformers -24layer	Fine-tuned	"[Value]"	MLE

Table 6: Data synthesis methods. "DCR" stands for Distance to the Closest Record and "MLE" stands for Machine Learning Efficiency.

[8]Available at `https://Github.com/RyanWangZf/MediTab`.
[9]The dataset is in `https://huggingface.co/datasets/yuweiyin/FinBench`
[10]The code is in `https://Github.com/YuweiYin/FinPT`

4.1 Methodologies

Borisov et al. (2023b) proposed GReaT[11] (Generation of Realistic Tabular data) to generate synthetic samples with original tabular data characteristics. The GReaT data pipeline involves a textual encoding step transforming tabular data into meaningful text using the sentences serialization methods as shown in Table 1, followed by fine-tuning GPT-2 or GPT-2 distill models. Additionally, a feature order permutation step precedes the use of obtained sentences for LLM fine-tuning.

REaLTabFormer (Solatorio & Dupriez, 2023) extends GReaT by generating synthetic non-relational and relational tabular data. It uses an autoregressive GPT-2 model to generate a parent table and a sequence-to-sequence model conditioned on the parent table for the relational dataset. The model implements target masking to prevent data copying and introduces statistical methods to detect overfitting. It demonstrates superior performance in capturing relational structures and achieves state-of-the-art results in predictive tasks without needing fine-tuning.

Following the similar paradigm, Zhang et al. (2023e) proposed the TAPTAP[12] (Table Pretraining for Tabular Prediction) which incorporates several enhancements. The method involves pre-fine-tuning the GPT2 on 450 Kaggle/UCI/OpenML tables, generating label columns using a machine learning model. Claimed improvements include a revised numerical encoding scheme and the use of external models like GBDT for pseudo-label generation, deviating from conventional language model-based approaches. However, the work lacks a comparison with diffusion-based models like TabDDPM, and the numerical encoding scheme improvement as highlighted in (Gruver et al., 2023) depends on the model used. In a related work (Wang et al., 2023a), a similar approach is employed for generating pseudo-labels, where the labels are represented as probability vectors.

TabuLa (Zhao et al., 2023f) addresses long training times of LLMs by advocating for a randomly initialized model as the starting point and shows the potential for continuous refinement through iterative fine-tuning on successive tabular data tasks [13]. It introduces a token sequence compression method and a middle padding strategy to simplify training data representation and enhance performance, achieving a significant reduction in training time while maintaining or improving synthetic data quality.

Seedat et al. (2023) introduces Curated LLM, a framework that leverages learning dynamics and two novel curation metrics, namely confidence and uncertainty. These metrics are employed to filter out undesirable generated samples during the training process of a classifier, aiming to produce high-quality synthetic data. Specifically, both metrics are calculated for each sample, utilizing the classifier trained on these samples. Additionally, CLLM distinguishes itself by not requiring any fine-tuning of LLMs, specifically utilizing the GPT-4.

TabMT (Gulati & Roysdon, 2023) employs a masked transformer-based architecture. The design allows efficient handling of various data types and supports missing data imputation. It leverages a masking mechanism to enhance privacy and data utility, ensuring a balance between data realism and privacy preservation. TabMT's architecture is scalable, making it suitable for diverse datasets and demonstrating improved performance in synthetic data generation tasks.

4.2 Evaluation

As outlined in Zhang et al. (2023c), the evaluation of synthetic data quality can be approached from four different dimensions: 1) **Low-order statistics** – *column-wise density* and *pair-wise column correlation*, estimating individual column density and the relational dynamics between pairs of columns, 2) **High-order metrics** – the calculation of *α-precision* and *β-recall* scores that measure the overall fidelity and diversity of synthetic data, 3) **privacy preservation** – DCR score, representing the median Distance to the Closest Record (DCR), to evaluate the privacy level of the original data, and 4) Performance on **downstream tasks** – like *machine learning efficiency* (MLE) and *missing value imputation*. MLE is to compare the testing accuracy on real data when trained on synthetically generated tabular datasets. Additionally, the

[11] The code is in `https://github.com/kathrinse/be_great`
[12] The code is in `https://github.com/ZhangTP1996/TapTap`
[13] The code is in `https://github.com/zhao-zilong/Tabula`

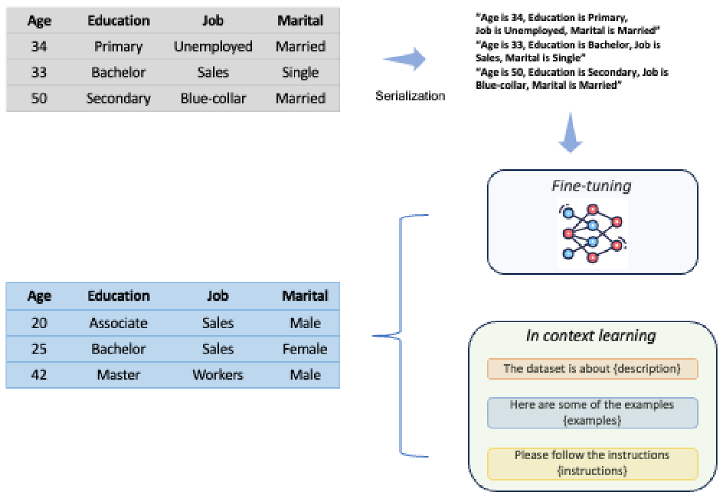

Figure 3: General data generation pipeline

quality of data generation can be assessed through its performance in the task of missing value imputation, which focuses on the replenishment of incomplete features/labels using available partial column data.

5 LLMs for question answering and table understanding

In this section, we cover datasets, trends and methods explored by researchers for question answering (QA), fact verification (FV) and table reasoning tasks. There are many papers working on database manipulation, management and integration (Lobo et al., 2023; Fernandez et al., 2023; Narayan et al., 2022; Zhang et al., 2023b), which also include instructions and tabular inputs to LLMs. However, they are not typically referred to as a QA task, and will not be covered by this paper.

5.1 Dataset

Table 7 outlines some of the popular datasets and benchmark in the literature working on tabular QA tasks.

Table QA For table QA datasets, we recommend to benchmark FetaQA (Nan et al., 2022) over WikiTable-Question (Pasupat & Liang, 2015a). Unlike WikiTableQuestions, which focuses on evaluating a QA system's ability to understand queries and retrieve short-form answers from tabular data, FeTaQA introduces elements that require deeper reasoning and integration of information. This includes generating free-form text answers that involve the retrieval, inference, and integration of multiple discontinuous facts from structured knowledge sources like tables. This requires the model generated long, informative, and free-form answers. NQ-TABLES Herzig et al. (2021) is larger than previously mentioned table. Its advantage lies in its emphasis on open-domain questions, which can be answered using structured table data. The code is in footnote [14].

Table and Conversation QA For QA task that involved both conversation and tables, we recommend to use HybriDialogue (Nakamura et al., 2022). HybriDialogue includes conversations grounded on both Wikipedia text and tables. This addresses a significant challenge in current dialogue systems: conversing on topics with information distributed across different modalities, specifically text and tables. The dataset is in footnote. [15]

[14]The dataset for NQ-Tables is in `https://github.com/google-research-datasets/natural-questions`. The dataset for WikiTableQuestions is in `https://ppasupat.github.io/WikiTableQuestions/`. The dataset for FetaQA is in `https://github.com/Yale-LILY/FeTaQA`.

[15]The dataset is in `https://github.com/entitize/HybridDialogue`

Dataset	# Tables	Task Type	Input	Output	Data Source	Papers Working on It
FetaQA Nan et al. (2022)	10330	QA	Table Question	Answer	Wikipedia	Ye et al. (2023b); Chen (2023); Sarkar & Lausen (2023); Zhao et al. (2023c)
WikiTableQuestion Pasupat & Liang (2015a)	2108	QA	Table Question	Answer	Wikipedia	Ye et al. (2023b); Chen (2023); Yin et al. (2020b); Jiang et al. (2023)
NQ-TABLES Herzig et al. (2021)	169898	QA	Question, Table	Answer	Synthetic	Chen et al. (2023a); Zhao et al. (2023c)
HybriDialogue Nakamura et al. (2022)	13000	QA	Conversation, Table, Reference	Answer	Wikipedia	?Sundar & Heck (2023); Zhang et al. (2023f); Zhao et al. (2023c)
TAT-QA Zhu et al. (2021a)	2757	QA	Question, Table	Answer	Financial report	Zhu et al. (2021a); Zhao et al. (2023c)
HiTAB Cheng et al. (2022)	3597	QA/NLG	Question, Table	Answer	Statistical Report and Wikipedia	Zhao et al. (2023a); Zhang et al. (2023f)
ToTTo Parikh et al. (2020a)	120000	NLG	Table	Sentence	Wikipedia	Sarkar & Lausen (2023); Zhang et al. (2023f)
FEVEROUS Aly et al. (2021)	28800	Classification	Claim, Table	Label	Wikipedia	Chen (2023); Sui et al. (2023c); Zhang et al. (2023f)
Dresden Web Tables Eberius et al. (2015)	125M	Classification	Table	Label	Common Crawl	Sarkar & Lausen (2023); Jin et al. (2023c)
InfoTabs Gupta et al. (2020)	2540	NLI	Table , Hypothesis	Label	Wikipedia	Akhtar et al. (2023); Yang et al. (2023)
TabFactChen et al. (2020a)	16573	NLI	Table, Statement	Label	Wikipedia	Zhang et al. (2023f); Jiang et al. (2023)
TAPEX Liu et al. (2022c)	1500	Text2SQL	SQL, Table	Answer	Synthetic	Sarkar & Lausen (2023); Yang et al. (2023)
Spider Yu et al. (2018b)	1020	Text2SQL	Table, Question	SQL	Human annotation	Yin et al. (2020b); Jiang et al. (2023)
WIKISQLZhong et al. (2017b)	24241	Text2SQL	Table, Question	SQL, Answer	Human Annotated	Chen et al. (2023a); Abraham et al. (2022); Zhang et al. (2023f); Jiang et al. (2023)

Table 7: Overview of Various Datasets and Related Work for LLMs for tabular QA data. We only select datasets that have been used by more than one relevant method in this table.

Table Classification We recommend to benchmark FEVEROUS Aly et al. (2021) if the tasks involve fact verification using both unstructural text and structured tables. We recommend to benchmark Dresden Web Tables (Eberius et al., 2015) for tasks requiring the classification of web table layouts, particularly useful in data extraction and web content analysis where table structures are crucial. The dataset is in footnote. [16]

Text2SQL If you want to create a SQL executor, you can use TAPEX (Liu et al., 2022c) and WIKISQL (Zhong et al., 2017b) which contains both tables , SQL query and answer. If you want to test ability to write a SQL query, you can use Spider (Yu et al., 2018b)[17], Magellan Das et al. or WIKISQL (Zhong et al., 2017b). Overall WIKISQL is preferable since it is large in size and has been benchmarked by many existed methods such as (Chen et al., 2023a; Abraham et al., 2022; Zhang et al., 2023f; Jiang et al., 2023) . The dataset is in footnote [18].

[16]The dataset for FEVEROUS is in `https://fever.ai/dataset/feverous.html`. The dataset for Dresden Web Tables is in `https://ppasupat.github.io/WikiTableQuestions/`.

[17]Leaderboard for Spider: `https://yale-lily.github.io/spider`

[18]The dataset for TAPEX is in `https://github.com/microsoft/Table-Pretraining/tree/main/data_generator`. The dataset for spider is in `https://drive.usercontent.google.com/download?id=1iRDVHLr4mX2wQKSgA9J8Pire73Jahh0m&export=download&authuser=0`. The dataset for WIKISQL is in `https://github.com/salesforce/WikiSQL`.

Table NLG ToTTo Parikh et al. (2020a) aims to create natural yet faithful descriptions to the source table. It is rich in size and can be used to benchmark table conditional text generation task. HiTAB (Cheng et al., 2022) allows for more standardized and comparable evaluation across different NLG models and tasks, potentially leading to more reliable and consistent benchmarking in the field. The dataset is in footnote. [19].

Table NLI InfoTabs (Gupta et al., 2020) uses Wikipedia infoboxes and is designed to facilitate understanding of semi-structured tabulated text, which involves comprehending both text fragments and their implicit relationships. InfoTabs is particularly useful for studying complex, multi-faceted reasoning over semi-structured, multi-domain, and heterogeneous data. TabFactChen et al. (2020a) consists of human-annotated natural language statements about Wikipedia tables. It requires linguistic reasoning and symbolic reasoning to get right answer. The dataset is in footnote. [20].

Domain Specific For airline industry specific table question answer, we recommend to use AIT-QA (Katsis et al., 2022). It highlights the unique challenges posed by domain-specific tables, such as complex layouts, hierarchical headers, and specialized terminology. For syntax description, we recommend to use TranX (Yin & Neubig, 2018). It uses an abstract syntax description language for the target representations, enabling high accuracy and generalizability across different types of meaning representations. For finance related table question answer, we recommend to use TAT-QA Zhu et al. (2021a). This dataset demands numerical reasoning for answer inference, involving operations like addition, subtraction, and comparison. Thus, TAT-QA can be used for complex task benchmark. The dataset is in footnote. [21].

Pretraining For pretraining on large datasets for table understanding, we recommend to use TaBERT (Yin et al., 2020c) and TAPAS (Herzig et al., 2020). Dataset in Tapas has 6.2 million tables and is useful for semantic parsing. TAPAS has 26 million tables and their associated english contexts. It can help model gain better understanding in both textual and table. The dataset is in footnote. [22].

5.2 General ability of LLMs in QA

Table 8 outlines the papers that investigated the effectiveness of LLMs on QA and reasoning, and the models explored. The most popular LLM used today is GPT3.5 and GPT4. Although these GPT models were not specifically optimized for table-based tasks, many of these papers found them to be competent in performing complex table reasoning tasks, especially when combined with prompt engineering tricks like CoT. In this section, we summarize the general findings of LLMs in QA tasks and highlight models that have reported to work well.

Numerical QA A niche QA task involves answering questions that require mathematical reasoning. An example query could be "*What is the average payment volume per transaction for American Express?*" Many real-world QA applications (E.g. working with financial documents, annual reports, etc.) involve such mathematical reasoning tasks. So far, Akhtar et al. (2023) conclude that LLMs like FlanT5 and GPT3.5 perform better than other models on various numerical reasoning tasks. On the DOCMATH-EVAL Zhao et al. (2023d) dataset, GPT-4 with CoT significantly outperforms other LLMs, while open-source LLMs (LLaMa-2, Vicuna, Mistral, Starcoder, MPT, Qwen, AquilaChat2, etc.) lag behind.

Text2SQL Liu et al. (2023c) designed a question matcher that identifies three keyword types: 1) column name-related terms, 2) restriction-related phrases (e.g. "top ten"), and 3) algorithm or module keywords. Once these keywords are identified, the module begins to merge the specific restrictions associated with each column into a unified combination, which is then matched with an SQL algorithm or module indicated by the third type of keyword. Zhang et al. (2023d) opted for a more straightforward approach of tasking LLaMa-2 to generate an SQL statement based on a question and table schema. Sun et al. (2023b) finetuned PaLM-2

[19]The dataset for ToTTo is in `https://github.com/google-research-datasets/ToTTo`. The dataset for HiTAB is in `https://github.com/microsoft/HiTab`

[20]The dataset for InfoTabs is in `https://infotabs.github.io/`. The dataset for TabFact is in `https://tabfact.github.io/`

[21]The dataset for AIT-QA is in `https://github.com/IBM/AITQA`. The dataset for TranX is in `https://github.com/pcyin/tranX`. The dataset for TAT-QA is in `https://github.com/NExTplusplus/TAT-QA`

[22]The dataset for TaBERT is in `https://github.com/facebookresearch/TaBERT`. The dataset for TAPAS is in `https://github.com/google-research/tapas`

Paper	Task	Models Explored
DOCMATH-EVAL (Zhao et al., 2023d)	NumQA	GPT4, GPT3.5, WizardLM, Llama-2 7, 13, 70B, CodeLlama 34B, Baichuan, Qwen, WizardMath, Vicuna, Mistral, etc.
Akhtar et al. (2023)	NumQA	TAPAS, DeBERTa, TAPEX, NT5, LUNA, PASTA, ReasTAP, FlanT5, GPT3.5, PaLM
TableGPT (Gong et al., 2020)	NumQA	GPT2
DATER (Ye et al., 2023b)	QA	GPT3 Codex
PACIFIC (Deng et al., 2022b)	QA	T5, CodeT5
Chen (2023)	QA	GPT3
cTBLS (Sundar & Heck, 2023)	QA	Custom: Dense Table Retrieval based on RoBERTa + Coarse State Tracking + Response based on GPT3.5
GPT4Table (Sui et al., 2023b)	QA	GPT-3.5, GPT-4
Zhao et al. (2023a)	QA	GPT-3.5
Liu et al. (2023e)	QA	GPT3.5
TableGPT (Zha et al., 2023)	QA	Phoenix-7B
TAP4LLM (Sui et al., 2023c)	QA	Instruct GPT3.5, GPT4
UniTabPT (Sarkar & Lausen, 2023)	QA	T5
Yu et al. (2023)	Multi-modal QA	Custom: Retrieval trained on contrastive loss, Rank by softmax, Generation built on T5
TableLlama (Zhang et al., 2023f)	QA	Custom: TableLlama
DIVKNOWQA Zhao et al. (2023c)	QA	GPT3.5, DSP, ReAct
Jiang et al. (2023)	QA	GPT3.5, ChatGPT3.5
Liu et al. (2023c)	QA & Text2SQL	Vicuna, GPT4
Gao et al. (2023)	Text2SQL	GPT4
Pourreza & Rafiei (2023)	Text2SQL	GPT4
Dong et al. (2023)	Text2SQL	ChatGPT3.5
Zhang et al. (2023d)	Text2SQL	LLaMA2 70b
Abraham et al. (2022)	Text2SQL	Custom: Table Selector + Known & Unknown Fields Extractor + AggFn Classifier

Table 8: Overview of Papers and Models for LLMs for tabular QA tasks. We only include papers that work with models of >1B parameters. Models that are described as "Custom" indicates papers that fine-tuned specific portions of their pipeline for the task, whereas the other papers focus more on non-finetuning methods like prompt engineering. NumQA: Numerical QA.

on the Text2SQL task, achieving considerable performance on Spider. The top scoring models for the Spider today are Dong et al. (2023); Gao et al. (2023); Pourreza & Rafiei (2023), all building off OpenAI's GPT models. SQL generation is popular in the industry, with many open-source fine-tuned models available.[23].

Impact of model size on performance Chen (2023) found that size does matter: On WebTableQuestions, when comparing the 6.7B vs. 175B GPT-3 model, the smaller model achieved only half the scores of the larger one. On TabFact, they found that smaller models (<=6.7B) obtained almost random accuracy.

Finetuning or No finetuning? Based on our survey, there is minimal work in the tabular QA space that finetunes LLMs (>70B parameters). This might be due to the general ability of LLMs (GPT3.5, GPT4) to perform many QA tasks without finetuning. For SQL generation on Spider, DIN-SQL Pourreza & Rafiei (2023) and DAIL-SQL are inference-based techniques using GPT4, and surpassed previous fine-tuned smaller models. The papers that finetune on QA based off smaller LLMs, are not the focus of this paper, and was mentioned previously in Section 2.1 under embeddings-based serialization. Instead, most papers working on tabular QA based on LLMs focus on the aspects of prompt engineering, search and retrieval, and end-to-end pipelines (user interfaces), which we describe further in the next section.

[23]`https://huggingface.co/NumbersStation`

5.3 Key components in QA

In the simplest QA architecture, an LLM takes in an input prompt (query and serialized table)[24], and returns an answer. In more involved architectures, the system might be connected to external databases or programs. Most of the times, the knowledge base might not fit in the context length or memory of the LLM. Therefore, unique challenges to tabular QA for LLMs include: query intent disambiguation, search and retrieval, output types and format, and multi-turn settings where iterative calls between programs are needed. We describe these components further in this section.

5.3.1 Query intent disambiguation

Zha et al. (2023) introduced the concept of Chain-of-command (CoC), that translates user inputs into a sequence of intermediate command operations. For example, an LLM needs to first check if the task requires retrieval, mathematical reasoning, table manipulations, and/or the questions cannot be answered if the instructions are too vague. They constructed a dataset of command chain instructions to fine-tune LLMs to generate these commands. Deng et al. (2022b) proposed the QA task be split into three subtasks: Clarification Need Prediction (CNP) to determine whether to ask a question for clarifying the uncertainty; Clarification Question Generation (CQG) to generate a clarification question as the response, if CNP detects the need for clarification; and Conversational Question Answering (CQA) to directly produce the answer as the response if it is not required for clarification. They trained a UniPCQA model which unifies all subtasks in QA through multi-task learning.

5.3.2 Search and retrieval

The ability to accurately search and retrieve information from specific positions within structured data is crucial for LLMs. There are two types of search and retrieval use-cases: (1) to find the information (table, column, row, cell) relevant to the question, and (2) to obtain additional information and examples.

For main table Zhao et al. (2023d) observed that better performance of a retriever module (that returns the top-n most relevant documents) consistently enhances the final accuracy of LLMs in numerical QA. Sui et al. (2023c) explored multiple table sampling methods (of rows and columns) and table packing (based on a token-limit parameter). The best technique was the query-based sampling, which retrieves rows with the highest semantic similarity to the question, surpassing methods involving no sampling, or clustering, random, even sampling, or content snapshots. Dong et al. (2023) used ChatGPT to rank tables based on their relevance to the question using SC: they generate ten sets of retrieval results, each set containing the top four tables, then selecting the set that appears most frequently among the ten sets. To further filter the columns, all columns are ranked by relevance to the question by specifying that ChatGPT match the column names against with the question words or the foreign key should be placed ahead to assist in more accurate recall results. Similarly, SC method is used. cTBLS Sundar & Heck (2023) designed a three-step architecture to retrieve and generate dialogue responses grounded on retrieved tabular information. In the first step, a dual-encoder-based Dense Table Retrieval (DTR) model, initialized from RoBERTa Liu et al. (2019), identifies the most relevant table for the query. In the second step, a Coarse System State Tracking system, trained using triplet loss, is used to rank cells. Finally, GPT-3.5 is prompted to generate a natural language response to a follow-up query conditioned on cells of the table ranked by their relevance to the query as obtained from the coarse state tracker. The prompt includes the dialogue history, ranked knowledge sources, and the query to be answered. Their method produced more coherent responses than previous methods, suggesting that improvements in table retrieval, knowledge retrieval, and response generation lead to better downstream performance. Zhao et al. (2023d) used OpenAI's Ada Embedding4 and Contriever (Izacard et al., 2022) as the dense retriever along with BM25 (Robertson et al., 1995) as the sparse retriever. These retrievers help to extract the top-n most related textual and tabular evidence from the source document, which were then provided as the input context to answer the question.

[24]For the scope of our paper, we do not consider images, videos and audio inputs.

For additional information Some papers explore techniques to curate samples for in-context learning. Gao et al. (2023) explored the a few methods: (1) random: randomly selecting k examples; (2) question similarity selection: choosing k examples based on semantic similarity with question Q, based on a predefined distance metric (E.g. Euclidean or negative cosine similarity) of the question and example embedding, and kNN algorithm to select k closest examples from Q; (3) masked question similarity selection: similar to (2), but beforehand masking domain-specific information (the table names, column names and values) in the question; (4) query similarity selection: select k examples similar to target SQL query $s*$, which relies on another model to generate SQL query s' based on the target question and database, and so s' is an approximation for $s*$. Output queries are encoded into binary discrete syntax vectors. Narayan et al. (2022) explored manually curated and random example selection.

5.3.3 Multi-turn tasks

Some papers design pipelines that call LLMs iteratively. We categorize the use-cases for doing so into three buckets: (1) to decompose a challenging task into manageable sub-tasks, (2) to update the model outputs based on new user inputs, and (3) to work-around specific constraints or to resolve errors.

Intermediate, sub-tasks This section overlaps with concepts around CoT and SC discussed earlier in Section 2.3. In a nutshell, since the reasoning task might be complex, LLMs might require guidance to decompose the task into manageable sub-tasks. For example, to improve downstream tabular reasoning, Sui et al. (2023b) proposed a two-step self-augmented prompting approach: first using prompts to ask the LLM to generate additional knowledge (intermediate output) about the table, then incorporating the response into the second prompt to request the final answer for a downstream task. Ye et al. (2023b) also guided the LLM to decompose a huge table into a small table, and to convert a complex question into simpler sub-questions for text reasoning. Their strategy achieved significantly better results than competitive baselines for table-based reasoning, outperforms human performance for the first time on the TabFact dataset. For Liu et al. (2023e), in encouraging symbolic CoT reasoning pathways, they allowed the model to interact with a Python shell that could execute commands, process data, and scrutinize results, particularly within a pandas dataframe, limited to a maximum of five iterative steps.

Dialogue-based applications In various applications where the users are interacting with the LLMs, like in chatbots, the pipeline must allow for LLMs to be called iteratively. Some dialogue-based Text2SQL datasets to consider are the SParC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a) datasets. For SParC, the authors designed subsequent follow-up questions based on Spider (Yu et al., 2018b).

Working around constraints or error de-bugging Zhao et al. (2023a) used multi-turn prompts to work around cases where the tables exceed the API input limit. In other cases, especially if the generated LLM output is code, an iterative process of feeding errors back to the LLM can help the LLM generate correct code. Zhang et al. (2023d) did so to improve SQL query generation.

5.3.4 Output evaluation and format

If the QA output is a number or category, F1 or Accuracy evaluation metrics are common. If evaluating open-ended responses, apart from using typical measures for like ROUGE and BLEU, some papers also hire annotators to evaluate the Informativeness, Coherence and Fluency of the LLM responses Zhang et al. (2023g). When connected to programs like Python, Power BI, etc, LLMs' outputs are not limited to text and code. For example, creating visualizations from text and table inputs are a popular task too Zhang et al. (2023g); Zha et al. (2023).

6 Limitations and future directions

LLMs has already been used in many tabular data applications, such as predictions, data synthesis, question answering and table understanding. Here we outline some practical limitations and considerations for future research.

Bias and fairness LLMs tend to inherit social biases from their training data, which significantly impact their fairness in tabular prediction and question answering tasks. Liu et al. (2023f) uses GPT3.5 and do few-shot learning to evaluate the fairness of tabular prediction on in context learning. The research concludes that LLMs tend to inherit social biases from their training data, which significantly impact their fairness in tabular prediction tasks. The fairness metric gap between different subgroups is still larger than that in traditional machine learning model. Additionally, the research further reveals that flipping the labels of the in-context examples significantly narrows the gap in fairness metrics across different subgroups, but comes at the expected cost of a reduction in predictive performance. The inherent bias of LLM is hard to mitigate through prompt (Hegselmann et al., 2023). Thus, a promising approach has proposed to mitigate bias is through pre-processing (Shah et al., 2020) or optimization (Bassi et al., 2024).

Hallucination LLMs have the risk of producing content that is inconsistent with the real-world facts or the user inputs (Huang et al., 2023). Hallucination raises concerns over the reliability and usefulness of LLMs in the real-world applications. For example, when working with patient records and medical data, hallucinations have critical consequences. Akhtar et al. (2023) found that hallucination led to performance drops in reasoning for LLMs. To address these issues, Wang et al. (2023c) incorporated an audit module that utilizes LLMs to perform self-check and self-correction. They generated pseudo-labels, then used a data audit module which filters the data based on data Shapley scores, leading to a smaller but cleaner dataset. Secondly, they also removed any cells with False values, which removes the chances of the LLMs making false inference on these invalid values. Finally, they performed a sanity check via LLM's reflection: They queried the LLM with the input template *"What is the {column}? {x}"* to check if the answer matches the original values. If the answers do not match, the descriptions are corrected by re-prompting the LLM. However, this method is far from efficient. Better methods to deal with hallucination could make LLMs' application in tabular data modeling more practical.

Numerical representation It was revealed that LLM in house embedding is not suitable for representing intrinsic relations in numerical features (Gruver et al., 2023), so specific embedding is needed. Tokenization significantly impacts pattern formation and operations in language models. Traditional methods like Byte Pair Encoding (BPE) used in GPT-3 often split numbers into non-aligned tokens (e.g., 42235630 into [422, 35, 630]), complicating arithmetic. Newer models like LLaMA tokenize each digit separately. Both approaches make LLM difficult to understand the whole number. Also, based on Spathis & Kawsar (2023), the tokenization of integers lacks a coherent decimal representation, leading to a fragmented approach where even basic mathematical operations require memorization rather than algorithmic processing. The development of new tokenizers, like those used in LLaMA (Touvron et al., 2023b), which outperformed GPT-4 in arithmetic tasks, involves rethinking tokenizer design to handle mixed textual and numerical data more effectively, such as by splitting each digit into individual tokens for consistent number tokenization (Gruver et al., 2023). This method has shown promise in improving the understanding of symbolic and numerical data. However, it hugely increases the dimension of the input which makes the method not practical for large datasets and many features.

Categorical representation Tabular dataset very often contains an excessive number of columns, which can lead to serialized input strings surpassing the context limit of the language model and increased cost. This is problematic as it results in parts of the data being pruned, thereby negatively impacting the model's performance. sample/truncate. Additionally, there are issues with poorly represented categorical feature, such as nonsensical characters, which the model struggles to process and understand effectively. Another concern is inadequate or ambiguous Metadata, characterized by unclear or meaningless column names and metadata, leading to confusion in the model's interpretation of inputs. Better categorical features encoding is needed to solve these problems.

Standard benchmark LLMs for tabular data could greatly benefit from standardized benchmark datasets to enable fair and transparent comparisons between models. In this survey, we strive to summarize commonly used datasets/metrics and provide recommendations for dataset selection to researchers and practitioners. However, the heterogeneity in tasks and datasets remains a significant challenge, hindering fair comparisons of model performance. Therefore, there is a pressing need for more standardized and unified datasets to bridge this gap effectively.

Model interpretability Like many deep learning algorithms, output from LLM suffers from a lack of interpretability. Only a few systems expose a justification of their model output such as TabLLM Hegselmann et al. (2023). One direction is to use the Shapley to derive interpretations. Shapley has been used to evaluate the prompt for LLM (Liu et al., 2023a). It could also be useful to understand how each feature influence the result. For instance, in prediction for diseases, providing explanation is crucial. In this case, a basic Shapley explanations would be able to show all features that led to the final decision. Future research is needed to explore the mechanisms for LLM's emerging capabilities for tabular data understanding.

Easy to use Currently, most relevant models require fine-tuning or data serialization, which could make these models hard to access. Some pretrained model such as Wang et al. (2023c); **?** could make people easy to use. It would be much easier to access if we can integrate these models with auto data prepossessing and serialization to existed platform such as Hugging Face.

Fine-tuning strategy design Designing appropriate tasks and learning strategies for LLMs is crucial. While LLMs demonstrate emergent abilities such as in-context learning, instruction following, and step-by-step reasoning, these capabilities may not be fully evident in certain tasks, depending on the model used. Also, LLMs are sensitive to various serialization and prompt engineering methods, which is the primary way to adapt LLM to unseen tasks. Thus, researchers and practitioners need to carefully design tasks and learning strategies tailored to specific models in order to achieve an optimal performance.

Model grafting The performance of LLM for tabular data modeling could be improved through model grafting. Model grafting involves mapping non-text data into the same token embedding space as text using specialized encoders, as exemplified by the HeLM model (Belyaeva et al., 2023), which integrates spirogram sequences and demographic data with text tokens. This approach is efficient and allows integration with high-performing models from various domains but adds complexity due to its non-end-to-end training nature and results in communication between components that is not human-readable. This approach could be adapted to LLM for tabular data to improve the encoding of non-text data.

7 Conclusion

This survey represents the first comprehensive investigation into the utilization of LLMs for modeling heterogeneous tabular data across various tasks, including prediction, data synthesis, question answering and table understanding. We delve into the essential steps required for tabular data to be ingested by LLM, covering serialization, table manipulation, and prompt engineering. Additionally, we systematically compare datasets, methodologies, metrics and models for each task, emphasizing the principal challenges and recent advancements in understanding, inferring, and generating tabular data. We provide recommendations for dataset and model selection tailored to specific tasks, aimed at aiding both ML researchers and practitioners in selecting appropriate solutions for tabular data modeling using different LLMs. Moreover, we examine the limitations of current approaches, such as susceptibility to hallucination, fairness concerns, data pre-processing intricacies, and result interpretability challenges. In light of these limitations, we discuss future directions that warrant further exploration in future research endeavors.

With the rapid development of LLMs and their impressive emergent capabilities, there is a growing demand for new ideas and research to explore their potential in modeling structured data for a variety of tasks. Through this comprehensive review, we hope it can provide interested readers with pertinent references and insightful perspectives, empowering them with the necessary tools and knowledge to effectively navigate and address the prevailing challenges in the field.

References

Abhijith Neil Abraham, Fariz Rahman, and Damanpreet Kaur. Tablequery: Querying tabular data with natural language. *CoRR*, abs/2202.00454, 2022. URL `https://arxiv.org/abs/2202.00454`.

Mubashara Akhtar, Abhilash Reddy Shankarampeta, Vivek Gupta, Arpit Patil, Oana Cocarascu, and Elena Simperl. Exploring the numerical reasoning capabilities of language models: A comprehensive analysis on tabular data. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for*

Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023, pp. 15391–15405. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.findings-emnlp.1028.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. FEVEROUS: fact extraction and verification over unstructured and structured information. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/68d30a9594728bc39aa24be94b319d21-Abstract-round1.html.

Sercan O. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning, 2020.

Gilbert Badaro, Mohammed Saeed, and Paolo Papotti. Transformers for tabular data representation: A survey of models and applications. *Transactions of the Association for Computational Linguistics*, 11: 227–249, 2023. doi: 10.1162/tacl_a_00544. URL https://aclanthology.org/2023.tacl-1.14.

Mrinal Kanti Baowaly, Chia-Ching Lin, Chao-Lin Liu, and Kuan-Ta Chen. Synthesizing electronic health records using improved generative adversarial networks. *Journal of the American Medical Informatics Association*, 26(3):228–241, 2019.

Pedro RAS Bassi, Sergio SJ Dertkigil, and Andrea Cavalli. Improving deep neural network generalization and robustness to background bias via layer-wise relevance propagation optimization. *Nature Communications*, 15(1):291, 2024.

Anastasiya Belyaeva, Justin Cosentino, Farhad Hormozdiari, Krish Eswaran, Shravya Shetty, Greg Corrado, Andrew Carroll, Cory Y McLean, and Nicholas A Furlotte. Multimodal llms for health grounded in individual-specific data. In *Workshop on Machine Learning for Multimodal Healthcare Data*, pp. 86–102. Springer, 2023.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. Gpt-neox-20b: An open-source autoregressive language model, 2022.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023a. URL https://openreview.net/pdf?id=cEygmQNOeI.

Vadim Borisov, Kathrin Sessler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=cEygmQNOeI.

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. Language models are few-shot learners, 2020.

Shaofeng Cai, Kaiping Zheng, Gang Chen, H. V. Jagadish, Beng Chin Ooi, and Meihui Zhang. Armnet: Adaptive relation modeling network for structured data. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD/PODS '21. ACM, June 2021. doi: 10.1145/3448016.3457321. URL `http://dx.doi.org/10.1145/3448016.3457321`.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, jan 2024. ISSN 2157-6904. doi: 10.1145/3641289. URL `https://doi.org/10.1145/3641289`.

Nuo Chen, Linjun Shou, Ming Gong, Jian Pei, Chenyu You, Jianhui Chang, Daxin Jiang, and Jia Li. Bridge the gap between language models and tabular understanding. *CoRR*, abs/2302.09302, 2023a. doi: 10.48550/ARXIV.2302.09302. URL `https://doi.org/10.48550/arXiv.2302.09302`.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL `https://doi.org/10.1145/2939672.2939785`.

Wenhu Chen. Large language models are few(1)-shot table reasoners. In Andreas Vlachos and Isabelle Augenstein (eds.), *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pp. 1090–1100. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EACL.83. URL `https://doi.org/10.18653/v1/2023.findings-eacl.83`.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact : A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, April 2020a.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *CoRR*, abs/2211.12588, 2022. doi: 10.48550/ARXIV.2211.12588. URL `https://doi.org/10.48550/arXiv.2211.12588`.

Zhihong Chen, Feng Jiang, Junying Chen, Tiannan Wang, Fei Yu, Guiming Chen, Hongbo Zhang, Juhao Liang, Chen Zhang, Zhiyi Zhang, Jianquan Li, Xiang Wan, Benyou Wang, and Haizhou Li. Phoenix: Democratizing chatgpt across languages. *CoRR*, abs/2304.10453, 2023b. doi: 10.48550/ARXIV.2304.10453. URL `https://doi.org/10.48550/arXiv.2304.10453`.

Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D. Davison. Table search using a deep contextualized language model. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20. ACM, July 2020b. doi: 10.1145/3397271.3401044. URL `http://dx.doi.org/10.1145/3397271.3401044`.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems, 2016.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. HiTab: A hierarchical table dataset for question answering and natural language generation. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1094–1110, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.78. URL `https://aclanthology.org/2022.acl-long.78`.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. Binding language models in symbolic languages. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL `https://openreview.net/pdf?id=lH1PV42cbF`.

Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. Generating multi-label discrete patient records using generative adversarial networks, 2018.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. Prototypical verbalizer for prompt-based few-shot tuning, 2022.

Sajad Darabi and Yotam Elor. Synthesising multi-modal minority samples for tabular data, 2021.

Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, Pradap Konda, Yash Govind, and Derek Paulsen. The magellan data repository. `https://sites.google.com/site/anhaidgroup/projects/data`.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. TURL: table understanding through representation learning. *SIGMOD Rec.*, 51(1):33–40, 2022a. doi: 10.1145/3542700.3542709. URL `https://doi.org/10.1145/3542700.3542709`.

Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. PACIFIC: towards proactive conversational question answering over tabular and textual data in finance. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 6970–6984. Association for Computational Linguistics, 2022b. doi: 10.18653/V1/2022.EMNLP-MAIN.469. URL `https://doi.org/10.18653/v1/2022.emnlp-main.469`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. In *Advances in Neural Information Processing Systems*, 2022.

Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, lu Chen, Jinshu Lin, and Dongfang Lou. C3: Zero-shot text-to-sql with chatgpt, 2023.

Julian Eberius, Katrin Braunschweig, Markus Hentsch, Maik Thiele, Ahmad Ahmadov, and Wolfgang Lehner. Building the dresden web table corpus: A classification approach. In Ioan Raicu, Omer F. Rana, and Rajkumar Buyya (eds.), *2nd IEEE/ACM International Symposium on Big Data Computing, BDC 2015, Limassol, Cyprus, December 7-10, 2015*, pp. 41–50. IEEE Computer Society, 2015. doi: 10.1109/BDC.2015.30. URL `https://doi.org/10.1109/BDC.2015.30`.

Raul Castro Fernandez, Aaron J. Elmore, Michael J. Franklin, Sanjay Krishnan, and Chenhao Tan. How large language models will disrupt data management. *Proc. VLDB Endow.*, 16(11):3302–3309, 2023. doi: 10.14778/3611479.3611527. URL `https://www.vldb.org/pvldb/vol16/p3302-fernandez.pdf`.

Hao Fu, Yao; Peng and Tushar Khot. How does gpt obtain its ability? tracing emergent abilities of language models to their sources. *Yao Fu's Notion*, Dec 2022. URL https://yaofu.notion.site/How-does-GPT-Obtain-its-Ability-Tracing-Emergent-Abilities-of-Language-Models-to-their-Sources-b9a57

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. Text-to-sql empowered by large language models: A benchmark evaluation, 2023.

Heng Gong, Yawei Sun, Xiaocheng Feng, Bing Qin, Wei Bi, Xiaojiang Liu, and Ting Liu. TableGPT: Few-shot table-to-text generation with table structure reconstruction and content matching. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 1978–1988, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.179. URL https://aclanthology.org/2020.coling-main.179.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data?, 2022.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*, 2023.

Manbir S Gulati and Paul F Roysdon. TabMT: Generating tabular data with masked transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=qs4swxtIAQ.

Cheng Guo and Felix Berkhahn. Entity embeddings of categorical variables, 2016.

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction, 2017.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. INFOTABS: Inference on tables as semi-structured data. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2309–2324, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.210. URL https://aclanthology.org/2020.acl-main.210.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David A. Sontag. Tabllm: Few-shot classification of tabular data with large language models. In Francisco J. R. Ruiz, Jennifer G. Dy, and Jan-Willem van de Meent (eds.), *International Conference on Artificial Intelligence and Statistics, 25-27 April 2023, Palau de Congressos, Valencia, Spain*, volume 206 of *Proceedings of Machine Learning Research*, pp. 5549–5581. PMLR, 2023. URL https://proceedings.mlr.press/v206/hegselmann23a.html.

Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45, 2022.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. Tapas: Weakly supervised table parsing via pre-training. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 4320–4333. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.ACL-MAIN.398. URL https://doi.org/10.18653/v1/2020.acl-main.398.

Jonathan Herzig, Thomas Mueller, Syrine Krichene, and Julian Eisenschlos. Open domain question answering over tables via dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 512–519, 2021.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *CoRR*, abs/2311.05232, 2023. doi: 10.48550/ARXIV.2311.05232. URL `https://doi.org/10.48550/arXiv.2311.05232`.

Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. Tabtransformer: Tabular data modeling using contextual embeddings, 2020.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. TABBIE: Pretrained representations of tabular data. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3446–3456, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.270. URL `https://aclanthology.org/2021.naacl-main.270`.

Sergei Ivanov and Liudmila Prokhorenkova. Boost then convolve: Gradient boosting meets graph neural networks, 2021.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022, 2022. URL `https://openreview.net/forum?id=jKN1pXi7b0`.

Sukriti Jaitly, Tanay Shah, Ashish Shugani, and Razik Singh Grewal. Towards better serialization of tabular data for few-shot classification with large language models, 2023.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. StructGPT: A general framework for large language model to reason over structured data. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9237–9251, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.574. URL `https://aclanthology.org/2023.emnlp-main.574`.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023a.

Ming Jin, Qingsong Wen, Yuxuan Liang, Chaoli Zhang, Siqiao Xue, Xue Wang, James Zhang, Yi Wang, Haifeng Chen, Xiaoli Li, Shirui Pan, Vincent S. Tseng, Yu Zheng, Lei Chen, and Hui Xiong. Large models for time series and spatio-temporal data: A survey and outlook. *arXiv preprint arXiv:2310.10196*, 2023b.

Nengzheng Jin, Joanna Siebert, Dongfang Li, and Qingcai Chen. A survey on table question answering: Recent advances, 2022.

Rihui Jin, Jianan Wang, Wei Tan, Yongrui Chen, Guilin Qi, and Wang Hao. TabPrompt: Graph-based pre-training and prompting for few-shot table understanding. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 7373–7383, Singapore, December 2023c. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.493. URL `https://aclanthology.org/2023.findings-emnlp.493`.

Alexia Jolicoeur-Martineau, Kilian Fatras, and Tal Kachman. Generating and imputing tabular data via diffusion and flow-based gradient-boosted trees, 2023.

Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka. Well-tuned simple nets excel on tabular datasets, 2021.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. AIT-QA: Question answering dataset over complex tables in the airline industry. In Anastassia Loukina, Rashmi Gangadharaiah, and Bonan Min (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pp. 305–314, Hybrid: Seattle, Washington + Online, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-industry.34. URL https://aclanthology.org/2022.naacl-industry.34.

Liran Katzir, Gal Elidan, and Ran El-Yaniv. Net-dnf: Effective deep modeling of tabular data. In *International conference on learning representations*, 2020.

Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu. Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 384–394, 2019a.

Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu. TabNN: A universal neural network solution for tabular data, 2019b. URL https://openreview.net/forum?id=r1eJssCqY7.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. Unifiedqa-v2: Stronger generalization via broader cross-format training. *arXiv preprint arXiv:2202.12359*, 2022.

Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. *arXiv preprint arXiv:2210.04018*, 2022a.

Jayoung Kim, Chaejeong Lee, Yehjin Shin, Sewon Park, Minjung Kim, Noseong Park, and Jihoon Cho. Sos: Score-based oversampling for tabular data. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 762–772, 2022b.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J. Inman. 1d convolutional neural networks and applications: A survey, 2019.

Jannik Kossen, Neil Band, Clare Lyle, Aidan N. Gomez, Tom Rainforth, and Yarin Gal. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning, 2022.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. Tabddpm: Modelling tabular data with diffusion models, 2022.

Chaejeong Lee, Jayoung Kim, and Noseong Park. Codi: Co-evolving contrastive diffusion models for mixed-type tabular synthesis. *arXiv preprint arXiv:2304.12654*, 2023.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, September 2019. ISSN 1367-4811. doi: 10.1093/bioinformatics/btz682. URL http://dx.doi.org/10.1093/bioinformatics/btz682.

Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. Ctrl: Connect collaborative and language model for ctr prediction, 2023.

Zheng Li, Yue Zhao, and Jialin Fu. Sync: A copula based framework for generating synthetic data from aggregated sources, 2020.

Guang Liu, Jie Yang, and Ledell Wu. Ptab: Using the pre-trained language model for modeling tabular data, 2022a.

Hanxi Liu, Xiaokai Mao, Haocheng Xia, Jian Lou, and Jinfei Liu. Prompt valuation based on shapley values. *CoRR*, abs/2312.15395, 2023a. doi: 10.48550/ARXIV.2312.15395. URL https://doi.org/10.48550/arXiv.2312.15395.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022b.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *CoRR*, abs/2307.03172, 2023b. doi: 10.48550/ARXIV.2307.03172. URL https://doi.org/10.48550/arXiv.2307.03172.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. TAPEX: table pre-training via learning a neural SQL executor. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022c. URL https://openreview.net/forum?id=O50443AsCP.

Shangching Liu, Shengkun Wang, Tsungyao Chang, Wenqi Lin, Chung-Wei Hsiung, Yi-Chen Hsieh, Yu-Ping Cheng, Sian-Hong Luo, and Jianwei Zhang. Jarvix: A LLM no code platform for tabular data analysis and optimization. In Mingxuan Wang and Imed Zitouni (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: EMNLP 2023 - Industry Track, Singapore, December 6-10, 2023*, pp. 622–630. Association for Computational Linguistics, 2023c. URL https://aclanthology.org/2023.emnlp-industry.59.

Tennison Liu, Zhaozhi Qian, Jeroen Berrevoets, and Mihaela van der Schaar. Goggle: Generative modelling for tabular data by learning relational structure. In *The Eleventh International Conference on Learning Representations*, 2023d.

Tianyang Liu, Fei Wang, and Muhao Chen. Rethinking tabular data understanding with large language models, 2023e.

Yanchen Liu, Srishti Gautam, Jiaqi Ma, and Himabindu Lakkaraju. Investigating the fairness of large language models for predictions on tabular data. *Short Version in NeurIPS 2023 Workshop on Socially Responsible Language Modelling Research*, 2023f.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Lin Zhao, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, 1(2):100017, September 2023g. ISSN 2950-1628. doi: 10.1016/j.metrad.2023.100017. URL http://dx.doi.org/10.1016/j.metrad.2023.100017.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Zhaocheng Liu, Qiang Liu, Haoli Zhang, and Yuntian Chen. Dnn2lr: Interpretation-inspired feature crossing for real-world tabular data, 2021.

Elita Lobo, Oktie Hassanzadeh, Nhan Pham, Nandana Mihindukulasooriya, Dharmashankar Subramanian, and Horst Samulowitz. Matching table metadata with business glossaries using large language models, 2023.

Hui Luan and Chin-Chung Tsai. A review of using machine learning approaches for precision education. *Educational Technology & Society*, 24(1):250–266, 2021.

Haoran Luo, Fan Cheng, Heng Yu, and Yuqi Yi. Sdtr: Soft decision tree regressor for tabular data. *IEEE Access*, 9:55999–56011, 2021.

Chao Ma, Sebastian Tschiatschek, José Miguel Hernández-Lobato, Richard Turner, and Cheng Zhang. Vaem: a deep generative model for heterogeneous mixed type data, 2020.

Hariharan Manikandan, Yiding Jiang, and J Zico Kolter. Language models are weak learners, 2023.

Ggaliwango Marvin, Nakayiza Hellen, Daudi Jjingo, and Joyce Nakatumba-Nabende. Prompt engineering in large language models. In *International Conference on Data Intelligence and Cognitive Informatics*, pp. 387–402. Springer, 2023.

Christopher McMaster, David FL Liew, and Douglas EV Pires. Adapting pretrained language models for solving tabular prediction problems in the electronic health record, 2023.

Kai Nakamura, Sharon Levy, Yi-Lin Tuan, Wenhu Chen, and William Yang Wang. HybriDialogue: An information-seeking dialogue dataset grounded on tabular and textual data. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 481–492, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.41. URL https://aclanthology.org/2022.findings-acl.41.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryscinski, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir R. Radev. Fetaqa: Free-form table question answering. *Trans. Assoc. Comput. Linguistics*, 10:35–49, 2022. doi: 10.1162/TACL_A_00446. URL https://doi.org/10.1162/tacl_a_00446.

Avanika Narayan, Ines Chami, Laurel J. Orr, and Christopher Ré. Can foundation models wrangle your data? *Proc. VLDB Endow.*, 16(4):738–746, 2022. doi: 10.14778/3574245.3574258. URL https://www.vldb.org/pvldb/vol16/p738-narayan.pdf.

Soma Onishi and Shoya Meguro. Rethinking data augmentation for tabular data in deep learning, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. ToTTo: A controlled table-to-text generation dataset. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1173–1186, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.89. URL https://aclanthology.org/2020.emnlp-main.89.

Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. Totto: A controlled table-to-text generation dataset. *CoRR*, abs/2004.14373, 2020b. URL https://arxiv.org/abs/2004.14373.

Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10): 1071–1083, June 2018. ISSN 2150-8097. doi: 10.14778/3231751.3231757. URL http://dx.doi.org/10.14778/3231751.3231757.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pp. 1470–1480. The Association for Computer Linguistics, 2015a. doi: 10.3115/V1/P15-1142. URL https://doi.org/10.3115/v1/p15-1142.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables, 2015b.

Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, 2016. doi: 10.1109/DSAA.2016.49.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237. Association for Computational Linguistics, 2018a. doi: 10.18653/v1/N18-1202. URL `https://aclanthology.org/N18-1202`.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018b.

Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data, 2019.

Mohammadreza Pourreza and Davood Rafiei. Din-sql: Decomposed in-context learning of text-to-sql with self-correction, 2023.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.

Stephen E. Robertson, Steve Walker, and Micheline Hancock-Beaulieu. Large test collection experiments on an operational, interactive system: Okapi at TREC. *Inf. Process. Manag.*, 31(3):345–360, 1995. doi: 10.1016/0306-4573(94)00051-4. URL `https://doi.org/10.1016/0306-4573(94)00051-4`.

Francesco Rundo, Francesca Trenta, Agatino Luigi di Stallo, and Sebastiano Battiato. Machine learning for quantitative finance applications: A survey. *Applied Sciences*, 9(24):5574, 2019.

Maria Sahakyan, Zeyar Aung, and Talal Rahwan. Explainable artificial intelligence for tabular data: A survey. *IEEE access*, 9:135392–135422, 2021.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization, 2021.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=9Vrb9D0WI4`.

Soumajyoti Sarkar and Leonard Lausen. Testing the limits of unified sequence to sequence LLM pretraining on diverse table data tasks. *CoRR*, abs/2310.00789, 2023. doi: 10.48550/ARXIV.2310.00789. URL `https://doi.org/10.48550/arXiv.2310.00789`.

Rick Sauber-Cole and Taghi M Khoshgoftaar. The use of generative adversarial networks to alleviate class imbalance in tabular data: a survey. *Journal of Big Data*, 9(1):98, 2022.

Lawrence Saul and Fernando Pereira. Aggregate and mixed-order markov models for statistical language processing, 1997.

Nabeel Seedat, Nicolas Huynh, Boris van Breugel, and Mihaela van der Schaar. Curated llm: Synergy of llms and data curation for tabular augmentation in ultra low-data regimes. *arXiv preprint arXiv:2312.12112*, 2023.

Deven Santosh Shah, H Andrew Schwartz, and Dirk Hovy. Predictive biases in natural language processing models: A conceptual framework and overview. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5248–5264, 2020.

Ira Shavitt and Eran Segal. Regularization learning networks: deep learning for tabular datasets. *Advances in Neural Information Processing Systems*, 31, 2018.

Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms. *CoRR*, abs/2310.10358, 2023. doi: 10.48550/ARXIV.2310.10358. URL `https://doi.org/10.48550/arXiv.2310.10358`.

Dylan Slack and Sameer Singh. Tablet: Learning from instructions for tabular data, 2023.

Aivin V Solatorio and Olivier Dupriez. Realtabformer: Generating realistic relational and tabular data using transformers. *arXiv preprint arXiv:2302.02041*, 2023.

Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C. Bayan Bruss, and Tom Goldstein. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training, 2021.

Dimitris Spathis and Fahim Kawsar. The first step is the hardest: Pitfalls of representing and tokenizing temporal data for large language models, 2023.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Evaluating and enhancing structural understanding capabilities of large language models on tables via input designs. *CoRR*, abs/2305.13062, 2023a. doi: 10.48550/ARXIV.2305.13062. URL `https://doi.org/10.48550/arXiv.2305.13062`.

Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Gpt4table: Can large language models understand structured table data? a benchmark and empirical study, 2023b.

Yuan Sui, Jiaru Zou, Mengyu Zhou, Xinyi He, Lun Du, Shi Han, and Dongmei Zhang. Tap4llm: Table provider on sampling, augmenting, and packing semi-structured data for large language model reasoning, 2023c.

Baohua Sun, Lin Yang, Wenhan Zhang, Michael Lin, Patrick Dong, Charles Young, and Jason Dong. Supertml: Two-dimensional word embedding for the precognition on structured tabular data, 2019.

Chenxi Sun, Yaliang Li, Hongyan Li, and Shenda Hong. Test: Text prototype aligned embedding to activate llm's ability for time series, 2023a.

Ruoxi Sun, Sercan Ö. Arik, Hootan Nakhost, Hanjun Dai, Rajarishi Sinha, Pengcheng Yin, and Tomas Pfister. Sql-palm: Improved large language model adaptation for text-to-sql. *CoRR*, abs/2306.00739, 2023b. doi: 10.48550/ARXIV.2306.00739. URL `https://doi.org/10.48550/arXiv.2306.00739`.

Anirudh S. Sundar and Larry Heck. cTBLS: Augmenting large language models with conversational tables. In Yun-Nung Chen and Abhinav Rastogi (eds.), *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pp. 59–70, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.nlp4convai-1.6. URL `https://aclanthology.org/2023.nlp4convai-1.6`.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Comput. Surv.*, 55(6):109:1–109:28, 2023a. doi: 10.1145/3530811. URL `https://doi.org/10.1145/3530811`.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. UL2: unifying language learning paradigms. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL `https://openreview.net/pdf?id=6ruVLB727MC`.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. Lamda: Language models for dialog applications, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023b. doi: 10.48550/ARXIV.2307.09288. URL `https://doi.org/10.48550/arXiv.2307.09288`.

Muhammad Umer, Muhammad Awais, and Muhammad Muzammul. Stock market prediction using machine learning (ml) algorithms. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 8(4):97–116, 2019.

L Vivek Harsha Vardhan and Stanley Kok. Generating privacy-preserving synthetic tabular data using oblivious variational autoencoders. In *Proceedings of the Workshop on Economics of Privacy and Data Labor at the 37 th International Conference on Machine Learning (ICML)*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017. URL `https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

Haifeng Wang, Jiwei Li, Hua Wu, Eduard Hovy, and Yu Sun. Pre-trained language models and their applications. *Engineering*, 2022a.

Ruiyu Wang, Zifeng Wang, and Jimeng Sun. Unipredict: Large language models are universal tabular predictors. *arXiv preprint arXiv:2310.03266*, 2023a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL `https://openreview.net/pdf?id=1PL1NIMMrw`.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks, 2022b.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. TUTA: tree-based transformers for generally structured table pre-training. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao (eds.), *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pp. 1780–1790. ACM, 2021. doi: 10.1145/3447548.3467434. URL `https://doi.org/10.1145/3447548.3467434`.

Zifeng Wang and Jimeng Sun. Transtab: Learning transferable tabular transformers across tables, 2022.

Zifeng Wang, Chufan Gao, Cao Xiao, and Jimeng Sun. Meditab: Scaling medical tabular data predictors via data consolidation, enrichment, and refinement, 2023c.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022a.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022, 2022b. URL `https://openreview.net/forum?id=yzkSU5zdwD`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022c. URL `http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan, 2019.

Weijie Xu, Wenxiang Hu, Fanyou Wu, and Srinivasan Sengamedu. Detime: Diffusion-enhanced topic modeling using encoder-decoder based llm. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-emnlp.606. URL `http://dx.doi.org/10.18653/v1/2023.findings-emnlp.606`.

Xiaojun Xu, Chang Liu, and Dawn Song. Sqlnet: Generating structured queries from natural language without reinforcement learning, 2017.

Hao Xue and Flora D Salim. Prompt-based time series forecasting: A new task and dataset. *arXiv preprint arXiv:2210.08964*, 2022.

Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig, and Thomas Dillig. Sqlizer: query synthesis from natural language. *Proc. ACM Program. Lang.*, 1(OOPSLA), oct 2017. doi: 10.1145/3133887. URL https://doi.org/10.1145/3133887.

Bohao Yang, Chen Tang, Kun Zhao, Chenghao Xiao, and Chenghua Lin. Effective distillation of table-based reasoning ability from llms, 2023.

Chao Ye, Guoshan Lu, Haobo Wang, Liyao Li, Sai Wu, Gang Chen, and Junbo Zhao. Ct-bert: Learning better tabular representations through cross-table pre-training, 2023a.

Yunhu Ye, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. Large language models are versatile decomposers: Decomposing evidence and questions for table-based reasoning. In Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (eds.), *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pp. 174–184. ACM, 2023b. doi: 10.1145/3539618.3591708. URL https://doi.org/10.1145/3539618.3591708.

Pengcheng Yin and Graham Neubig. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In Eduardo Blanco and Wei Lu (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 7–12, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2002. URL https://aclanthology.org/D18-2002.

Pengcheng Yin, Graham Neubig, Wen tau Yih, and Sebastian Riedel. Tabert: Pretraining for joint understanding of textual and tabular data, 2020a.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. Tabert: Pretraining for joint understanding of textual and tabular data. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 8413–8426. Association for Computational Linguistics, 2020b. doi: 10.18653/V1/2020.ACL-MAIN.745. URL https://doi.org/10.18653/v1/2020.acl-main.745.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8413–8426, Online, July 2020c. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.745. URL https://aclanthology.org/2020.acl-main.745.

Yuwei Yin, Yazheng Yang, Jian Yang, and Qi Liu. Finpt: Financial risk prediction with profile tuning on pretrained foundation models, 2023.

Bowen Yu, Cheng Fu, Haiyang Yu, Fei Huang, and Yongbin Li. Unified language representation for question answering over text, tables, and images. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 4756–4765. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-ACL.292. URL https://doi.org/10.18653/v1/2023.findings-acl.292.

Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. Typesql: Knowledge-based type-aware neural text-to-sql generation, 2018a.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3911–3921, Brussels, Belgium, October-November 2018b. Association for Computational Linguistics. doi: 10.18653/v1/D18-1425. URL https://aclanthology.org/D18-1425.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander R. Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter S. Lasecki, and Dragomir R. Radev. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pp. 1962–1979. Association for Computational Linguistics, 2019a. doi: 10.18653/V1/D19-1204. URL https://doi.org/10.18653/v1/D19-1204.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir R. Radev. Sparc: Cross-domain semantic parsing in context. In Anna Korhonen, David R. Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 4511–4523. Association for Computational Linguistics, 2019b. doi: 10.18653/V1/P19-1443. URL https://doi.org/10.18653/v1/p19-1443.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Peng Zhang, Yuxiao Dong, and Jie Tang. Glm-130b: An open bilingual pre-trained model, 2023.

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. Tablegpt: Towards unifying tables, nature language and commands into one GPT. *CoRR*, abs/2307.08674, 2023. doi: 10.48550/ARXIV.2307.08674. URL https://doi.org/10.48550/arXiv.2307.08674.

Han Zhang, Xumeng Wen, Shun Zheng, Wei Xu, and Jiang Bian. Towards foundation models for learning on tabular data, 2023a.

Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. Jellyfish: A large language model for data preprocessing. *CoRR*, abs/2312.01678, 2023b. doi: 10.48550/ARXIV.2312.01678. URL https://doi.org/10.48550/arXiv.2312.01678.

Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. *arXiv preprint arXiv:2310.09656*, 2023c.

Hengyuan Zhang, Peng Chang, and Zongcheng Ji. Bridging the gap: Deciphering tabular data using large language model. *CoRR*, abs/2308.11891, 2023d. doi: 10.48550/ARXIV.2308.11891. URL https://doi.org/10.48550/arXiv.2308.11891.

Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 42(4), oct 2017. ISSN 0362-5915. doi: 10.1145/3134428. URL https://doi.org/10.1145/3134428.

Tianping Zhang, Shaowen Wang, Shuicheng Yan, Li Jian, and Qian Liu. Generative table pre-training empowers models for tabular prediction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14836–14854. Association for Computational Linguistics, December 2023e. doi: 10.18653/v1/2023.emnlp-main.917. URL https://aclanthology.org/2023.emnlp-main.917.

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

Weixu Zhang, Yifei Wang, Yuanfeng Song, Victor Junqiu Wei, Yuxing Tian, Yiyan Qi, Jonathan H. Chan, Raymond Chi-Wing Wong, and Haiqin Yang. Natural language interfaces for tabular data querying and visualization: A survey. *CoRR*, abs/2310.17894, 2023g. doi: 10.48550/ARXIV.2310.17894. URL `https://doi.org/10.48550/arXiv.2310.17894`.

Bowen Zhao, Changkai Ji, Yuejie Zhang, Wen He, Yingwen Wang, Qing Wang, Rui Feng, and Xiaobo Zhang. Large language models are complex table parsers. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 14786–14802. Association for Computational Linguistics, 2023a. URL `https://aclanthology.org/2023.emnlp-main.914`.

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. A survey of large language models, 2023b.

Wenting Zhao, Ye Liu, Tong Niu, Yao Wan, Philip S. Yu, Shafiq Joty, Yingbo Zhou, and Semih Yavuz. Divknowqa: Assessing the reasoning ability of llms via open-domain question answering over knowledge base and text, 2023c.

Yilun Zhao, Yitao Long, Hongjun Liu, Linyong Nan, Lyuhao Chen, Ryo Kamoi, Yixin Liu, Xiangru Tang, Rui Zhang, and Arman Cohan. Docmath-eval: Evaluating numerical reasoning capabilities of llms in understanding long documents with tabular data. *CoRR*, abs/2311.09805, 2023d. doi: 10.48550/ARXIV.2311.09805. URL `https://doi.org/10.48550/arXiv.2311.09805`.

Yilun Zhao, Chen Zhao, Linyong Nan, Zhenting Qi, Wenlin Zhang, Xiangru Tang, Boyu Mi, and Dragomir Radev. RobuT: A systematic study of table QA robustness against human-annotated adversarial perturbations. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6064–6081, Toronto, Canada, July 2023e. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.334. URL `https://aclanthology.org/2023.acl-long.334`.

Zilong Zhao, Robert Birke, and Lydia Chen. Tabula: Harnessing language models for tabular data synthesis. *arXiv preprint arXiv:2310.12746*, 2023f.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning, 2017a.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 2017b.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3277–3287, Online, August 2021a. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.254. URL `https://aclanthology.org/2021.acl-long.254`.

Yitan Zhu, Thomas Brettin, Fangfang Xia, Alexander Partin, Maulik Shukla, Hyunseung Yoo, Yvonne A Evrard, James H Doroshow, and Rick L Stevens. Converting tabular data into images for deep learning with convolutional neural networks. *Scientific reports*, 11(1):11325, 2021b.