Empowering Tabular Data Preparation with Language Models: Why and How?

Anonymous ARR submission

Abstract

Data preparation is a critical step in enhancing the usability of tabular data and thus boosts downstream data-driven tasks. Traditional methods often face challenges in capturing the intricate relationships within tables and adapting to the tasks involved. Recent advances in Language Models (LMs), especially in Large Language Models (LLMs), offer new opportunities to automate and support tabular data preparation. However, why LMs suit tabular data preparation (i.e. how their capabilities match task demands) and how to use them effectively across phases still remain to be systematically explored. In this survey, we systematically analyze the role of LMs in enhancing tabular data preparation processes, focusing on four core phases: data acquisition, integration, cleaning, and transformation. For each phase, we present an integrated analysis of how LMs can be combined with other components for different preparation tasks, highlight key advancements, and outline prospective pipelines.

1 Introduction

Tabular data preparation is the end-to-end procedure of transforming raw, heterogeneous tables into a clean, integrated, and analysis-ready form to enhance data usability. Effective tabular data preparation is critical for developing data-driven models, since it prevents the models from falling prey to the garbage in, garbage out (GIGO) syndrome (Brownlee, 2020; Jain et al., 2020; Stieglitz et al., 2018; Hameed and Naumann, 2020a; Jassim and Abdulwahid, 2021). Moreover, data preparation demands expert knowledge and involves a substantial labor burden, with data scientists reportedly spending up to 80% of their time on this task (Hameed and Naumann, 2020b). Given these challenges, numerous approaches are proposed to automate and support effective tabular data preparation.

In the early stages, tabular data preparation methods primarily relied on rule-based systems or traditional learning architectures such as tree-based models (Yohannes and Hoddinott, 1999; Friedman, 2001) and shallow networks (Kramer, 1991; Hawkins et al., 2002; Batista and Monard, 2003). However, these approaches struggle to comprehend the implicit relationships within tables and are difficult to adapt to complex tasks involved in tabular data preparation. Firstly, they often fail to model fine-grained semantic dependencies, which limits their effectiveness. Secondly, these methods are typically task-specific and designed with restrictive priors, which restrict their generalization capacity.

Recently, with strong capabilities in semantic understanding, language models (LMs), especially large language models (LLMs) have shown great promise for tabular data preparation (Naveed et al., 2023; Xi et al., 2023; Liu et al., 2023). Despite growing interest and notable advancements, existing literature on LM-enabled tabular data preparation often concentrates on isolated tasks or specific technical approaches, thereby lacking a systematic and comprehensive understanding of why LMs are suitable for addressing the inherent complexity of tabular data and how they can be most effectively employed (Lu et al., 2025; Li et al., 2024a; Ding et al., 2024a; Long et al., 2024; Zhu et al., 2024).

To address this critical gap, our survey provides a systematic exploration of **why** and **how** LMs can empower tabular data preparation. As shown in Figure 1, we structure this survey around four core and closely related tabular data preparation phases: (1) **data acquisition** that collects data from diverse sources. We focus on *table discovery*, *joinable table search* and *unionable table search*; (2) **data integration** that combines and aligns data. We focus on *schema matching* and *entity matching*; (3) **data cleaning** that handles missing values and errors. We focus on *error detection*, *data repair* and *data imputation*; (4) **data transformation** that formats and encodes data for analysis. We focus on *format transformation* and *semantic transformation*.

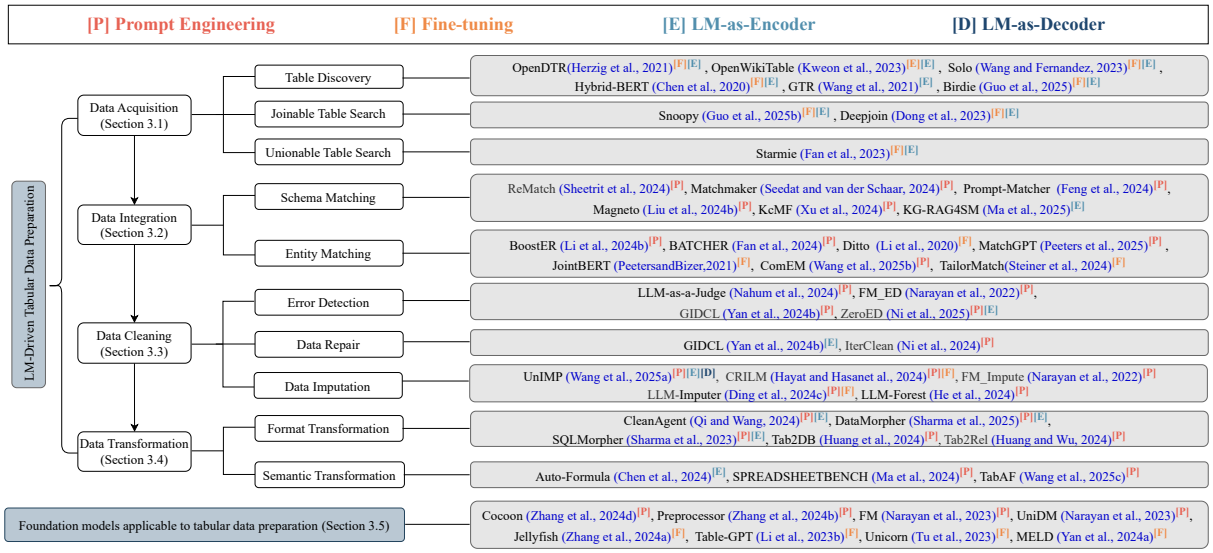

Figure 1: Illustration of the tabular data preparation pipeline, consisting of: data acquisition, data integration, data cleaning and data transformation. Foundation models applicable to data preparation are also explored.

For each of these phases, We first explore the **"Why"** by analyzing the nature of each task and the strengths of LMs that make them suitable for specific tasks. We then dissect the **"How"** by examining the two main enabling strategies through which these models are employed: (1) LM-centric strategies (e.g., prompt-based and fine-tuning-based) and (2) LM-in-the-loop strategies (e.g., LM-as-encoder and LM-as-decoder), in order to distill generalizable patterns across tasks and frameworks. While LMs show potential for tabular data preparation, the field is in its early stages. We summarize existing enabling approaches for each phase, pros and cons for different frameworks and highlight promising directions for future research.

2 Preliminaries

This paper investigates the use of LMs, including LLMs and small language models (SLMs), for tabular data preparation, which transforms raw tables into high-quality, analysis-ready formats. The detailed introduction of LLM and SLMs is provided in the Appendix A. Tabular data preparation is not a fixed pipeline but is often data-dependent, involving a set of operations to enhance quality.

Let $\mathcal{D}_{\text{lake}}$ denote the initial data lake, containing diverse tables from various sources. A preparation operator set is defined as: $\mathcal{O}_{\text{prep}} = \{O_{\text{acq}}, O_{\text{int}}, O_{\text{clean}}, O_{\text{trans}}, \ldots\}$ where O_{acq} represents data acquisition, O_{int} represents data integration, O_{clean} is data cleaning, and O_{trans} is data transformation. There may be other operators, but these four operators form the core of tabular data preparation and are the main focus of this paper.

We denote the state of the dataset at time step k as $\mathcal{D}^{(k)}$. At the k-th step, the system selects an operator $O_k \in \mathcal{O}_{\text{prep}}$ based on current data characteristics, either through expert knowledge (Zhang et al., 2003; Mining, 2006) or algorithms (Krishnan and Wu, 2019; Li et al., 2021). The selected operator is then applied to the $\mathcal{D}^{(k)}$:

$$\mathcal{D}^{(k+1)} \leftarrow \mathcal{M}^{O_k}\left(\mathcal{D}^{(k)}\right), \quad O_k \in \mathcal{O}_{\text{prep}},$$

where $\mathcal{M}^{O_k}$ represents an LM-based executor of operator O_k. The process continues until the dataset satisfies certain quality criteria or a predefined utility threshold is reached.

3 Tabular Data Preparation Process

3.1 Overview

Task Overview. As shown in Figure 1, tabular data preparation is typically comprising data acquisition, integration, cleaning, and transformation. In data acquisition, common tasks include table discovery, joinable table search, and unionable table search. Data integration focuses on schema matching and entity matching to align heterogeneous data sources. Data cleaning addresses error detection, data repair, and imputation to improve data quality. Finally, data transformation covers format transformation and semantic transformation to adapt data for downstream analytics. Definitions for all specific tasks can be found in the Appendix B.

Framework Overview. As shown in Figure 2, LM-based methods can be grouped into two main strategies: LM-centric strategies and LM-in-the-loop

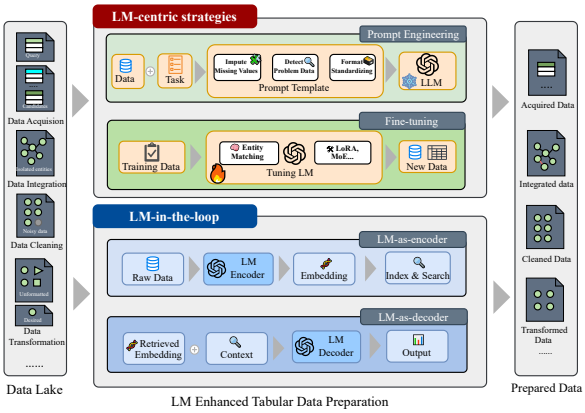

Figure 2: Framework of applying LMs for tabular data preparation: LM-centric (prompt engineering and fine-tuning) and LM-in-the-loop methods (LM-as-encoder and LM-as-decoder).

strategies. LM-centric strategies use the model directly, either through prompt engineering with curated instructions or by fine-tuning it on task-specific data. In contrast, LM-in-the-loop strategies integrate LLMs or SLMs with other components to combine complementary strengths. LMs usually function as encoders to produce semantic representations or as decoders to generate textual outputs.

Table 1 presents a taxonomy of LM-enabled tabular data preparation methods.

3.2 Data Acquisition

Touchpoint. Data acquisition is responsible for identifying relevant tables from large data lakes to support subsequent integration, cleaning, and transformation. It is often query-driven, where the acquisition process is guided by natural language inputs or query tables. Consequently, state-of-the-art methods typically formulate the problem as a similarity search between the query and data within the data lake, leveraging the semantic understanding capabilities of LMs by incorporating them in an **LM-as-encoder** paradigm. Such capability enables LLMs to capture fine-grained semantic correspondences beyond lexical overlap. In this paper, we focus on three core tasks: table discovery, joinable table search and unionable table search.

3.2.1 Table Discovery

Table discovery aims to retrieve the top-k tables from a table repository that are most semantically relevant to a given natural language query. Early methods (Sayyadian et al., 2007; Sarma et al., 2012) relied on keyword-based search over table metadata and content, which struggled to capture semantic relevance beyond lexical matches. By employing LM as an encoder,with strong semantic understanding, table discovery can be specifically categorized into two approaches according to the embedding strategy: representation-index-search and rerank.

Representation-Index-Search. This approach typically follows a three-stage pipeline: (i) encoding query and data lake tables separately into embeddings, (ii) constructing indexes over these embeddings in an offline phase, and (iii) retrieving the top-k most relevant tables via similarity search. Variations across existing methods generally lie in the choice of embedding methods and indexing techniques. OpenDTR (Herzig et al., 2021) adopts TAPAS (Herzig et al., 2020) as the backbone of its encoder, while OpenWikiTable (Kweon et al., 2023) provides several encoder options such as BERT (Devlin et al., 2019a) and TAPAS. Solo (Wang and Fernandez, 2023) introduces a more fine-grained representation by encoding each cell–attribute–cell triplet into a fixed-dimensional embedding. For indexing, most methods rely on approximate nearest neighbor (ANN) algorithms, such as IVF-PQ (Johnson et al., 2021) in Solo (Wang and Fernandez, 2023). To further enhance the integration between indexing and retrieval, Birdie (Guo et al., 2025a) proposes a differentiable search index that directly determines.

Rerank. In contrast to methods that encode the query and data separately, the rerank approach leverages LLMs/SLMs to jointly encode the query and candidate tables, capturing fine-grained semantic interactions. As it requires computing new embeddings for each query, this approach is generally more time-consuming. To mitigate this, a lightweight retriever first selects a pool of candidate tables, which are then reranked by a joint encoder based on query–table interactions. Hybrid-BERT (Chen et al., 2020) employs a content selector to extract relevant table segments and uses BERT with a regression layer to predict matching scores. GTR (Wang et al., 2021) introduces a pooling mechanism to integrate the embeddings of the query and candidate table.

Pros & Cons. Representation-Index-Search approach is highly scalable and efficient but may lose fine-grained alignment information and is less convenient to update when new data is added. Rerank excels at capturing the correlation between a query and each candidate table by joint encoding, which improves semantic precision. However, it can be

computationally intensive and less scalable.

3.2.2 Joinable Table Search

Joinable table search seeks to identify the top-k tables from a collection that contain a column capable of serving as a join key with a query table. Classic approaches tackled this through syntax-based candidate filtering methods like LSH-based blocking (Zhu et al., 2016) or explored structural links using graph-based methods (Koutras et al., 2024).

Recently, LM-as-encoder is explored to capture semantic information to improve performance. Similar to table discovery, these methods often follow a representation–index–search paradigm. DeepJoin (Dong et al., 2023) fine-tunes SLM-based encoders (e.g., DistilBERT (Sanh et al., 2019), MPNet (Song et al., 2020)) to capture the latent semantics of columns. It embeds the query column and retrieves joinable candidates from a prebuilt HNSW index (Malkov and Yashunin, 2020). Snoopy (Guo et al., 2025b) also follows this paradigm and further introduces the technique of proxy columns to enhance representations by contrastive learning.

Pros & Cons. LM-as-encode models latent relationships and contextual meanings, which improves match quality but comes with higher computational cost and a risk of semantic drift.

3.2.3 Unionable Table Search

Unionable table search aims to retrieve the top-k tables from a collection that can be vertically concatenated with a given query table.

In recent years, the representation–index–search paradigm has been developed in parallel with the LM-as-encoder approach to capture contextual information across multiple columns. Starmie (Fan et al., 2023) represents tables and reasons about schema-level similarity using an SLM (RoBERTa (Liu et al., 2019)). It encodes both the query and candidate tables holistically, capturing inter-column relationships. It then employs HNSW (Malkov and Yashunin, 2020) for efficient indexing and introduces bipartite matching to compute table-level similarity scores while enabling pruning during online search.

Pros & Cons. Table-based methods consider the whole table context and capture richer relationships, but may incur higher computational and memory costs and may overlook fine-grained information.

Summary and take-away. LLMs/SLMs for data acquisition are typically used as encoders, framing the task as a similarity search. They are often integrated into pipelines following a representation–index–search or re-ranking paradigm. Most existing approaches focus on leveraging SLMs to enhance efficiency.

3.3 Data Integration

Touchpoint. Following data acquisition, data integration focuses on aligning heterogeneous and inconsistent tables to enable subsequent phases. Unlike data acquisition, data integration does not involve explicit queries. Instead, every table can be considered a query that needs to be compared with all other tables, resulting in a quadratic complexity. Hence, state-of-the-art approaches typically **prompt a LLM** as a reranker or matcher to leverage its deep semantic understanding, allowing the model to reason over heterogeneous tables and generalize beyond explicit column matches. To improve efficiency, recent approaches incorporate SLMs and rule-based heuristics to filter candidats before invoking the LLM. In this survey, we focus on surveying schema matching and entity matching.

3.3.1 Schema Matching

Schema matching aims to learn a function that aligns attributes from a source schema with corresponding attributes in a target schema. Traditional methods have approached this either statistically (Castro Fernandez et al., 2018), or through machine learning (Koutras et al., 2021).

Prompt-based LLM reranker. Related methods usually first generate candidate matches, then design prompts with instructions and candidate matches to further refine them using LLMs (Parciak et al., 2024). They vary in the (1) candidate match generation process and (2) prompt designs.

For the candidate generation, ReMatch (Sheetrit et al., 2024) converts target schemas into retrievable documents, using semantic similarity to filter potential matches. KG-RAG4SM (Ma et al., 2025) leverages external knowledge graphs, encoding the information from graph-structured data and external knowledge graphs for semantic alignment. Prompt-Matcher (Feng et al., 2024) employs traditional schema-matching methods to produce probabilistic candidates. Magneto (Liu et al., 2024b) adopts a lightweight SLM for candidate retrieval.

Different prompt designs are also explored to better utilize LLM as a rerank. Matchmaker (Seedat

and van der Schaar, 2024) iteratively refines candidate matches using confidence scores and multi-round prompt feedback. KcMF (Xu et al., 2024) injects domain knowledge and pseudo-code-like instructions into prompts, explicitly guiding LLMs to reduce confusion and hallucinations.

Pros & Cons. Prompt-based LLM rerankers leverage small schema sizes and pre-filtering to enable efficient reranking. They can also use rich contextual cues to resolve ambiguous matches, improving precision. However, poor initial recall of candidates can cause valid matches to be missed.

3.3.2 Entity Matching

Entity matching identifies whether two records from different sets refer to the same entity. Traditional methods include deterministic rules, statistical methods, and machine learning methods (Binette and Steorts, 2020). Recently, following the widely adopted block-and-match framework (Wu et al., 2023), LMs are explored as a matcher by prompt engineering and fine-tuning.

Prompt Engineering. The method in (Peeters et al., 2025) evaluates various prompts for entity matching and shows that even a simple zero-shot prompt performs well. ComEM (Wang et al., 2025b) expands this paradigm by comparing multiple prompting styles: pairwise matching, side-by-side comparison, and selecting one correct record from a set. BATCHER (Fan et al., 2024b) improves cost-effectiveness by batching entity pairs and selecting in-context examples, significantly reducing token usage with little accuracy loss. BoostER (Li et al., 2024b) adopts selective verification, generating candidate matches with probabilities and refining them using Bayesian adjustment.

Fine-tuning. Fine-tuning is widely used for entity matching. In the early stage, most methods are primarily based on fine-tuning SLMs. JointBERT(Peeters and Bizer, 2021) fine-tunes BERT simultaneously on binary matching decisions and multi-class entity classification tasks, leveraging richer supervision signals. Similarly, Ditto(Li et al., 2020) demonstrates further gains by enhancing pre-trained transformer models (e.g., BERT, RoBERTa) with targeted textual augmentations and domain-specific knowledge injection. Method in (Steiner et al., 2024) finetune both SLM and LLM and show that properly fine-tuned smaller models consistently improve matching performance, whereas results for LLMs are more variable.

Pros & Cons. Prompt engineering enables the entity matching models to adapt quickly across different domains, without additional training. However, matching accuracy can be unstable when prompts fail to capture fine-grained differences between ambiguous entities. Fine-tuning improves alignment consistency and task-specific accuracy, but requires labeled pairs and retraining.

Summary and take-away. Leveraging the semantic capacity of LMs, both prompting and fine-tuning methods have been explored for data integration. To address efficiency concerns, LLM is often used as a reranker in the retrieval–rerank pipeline or a matcher in the blocker–matcher pipeline.

3.4 Data Cleaning

Touchpoint. Data cleaning aims to identify (error detection) and rectify (data repair and data imputation (Wang et al., 2024a)) errors or inconsistencies in a dataset to improve data quality. Unlike query-driven data acquisition and pairwise-oriented data integration, data cleaning is typically performed at the instance level. This prior offers increased flexibility and minimal structural constraints, allowing various strategies to be contextually incorporated into the cleaning process. Early methods based on SLMs offered efficient but limited support while recent advances leverage LLMs to enable more powerful detection, correction and imputation, benefiting from their ability to model dependencies and generate semantically consistent operations.

3.4.1 Error Detection

Error detection aims to identify erroneous cells on the table based on criteria such as integrity constraint violations and semantic inconsistencies. Error detection is a discriminative task. LM-based methods typically adopt two strategies: (1) using prompts to reframe the task as generative to leverage LLM knowledge, and (2) using LM-as-encoder to extract features for classifier training.

Prompt-based. Prompt-based methods have various implementation strategies. Some methods directly prompt the LLM to determine whether a given instance is erroneous (Narayan et al., 2022). To accelerate error detection, ZeroED (Ni et al., 2025) prompts the LLM to generate rules, which can be efficiently and scalably applied across large datasets. Furthermore, GIDCL (Yan et al., 2024b) adopts an iterative approach that leverages LLMs to generate rules for corrupting clean data and synthe-

sizing dirty samples. The LLM-as-a-Judge framework (Nahum et al., 2024) employs an ensemble of LLMs to detect errors in labels in the datasets.

LM-as-encoder. In the early stage, SLMs are used to encode data features and train detection models (e.g., Tabreformer (Nashaat et al., 2021)), balancing efficiency and accuracy. Recently, LLMs are explored. ZeroED (Ni et al., 2025) proposes LLM-based error reason-aware features, which are then used to train a binary classifier.

Pros & Cons. Prompt-based methods enable flexible rule induction and can handle diverse error types without task-specific training, but results may be inconsistent and sensitive to prompt formulation. LM-as-encoder approaches provide stable, reusable representations for large-scale detection, yet may struggle to capture rare or subtle errors.

3.4.2 Data Repair

Data repair seeks to rectify detected erroneous cells in a raw dataset to construct a new dataset free from such errors. Different from error detection, which is discriminative, data repair is essentially a generative task and thus well suited to LMs.

Prompt-based. IterClean (Ni et al., 2024) enhances this by providing task instructions along with detailed explanations for each identified error to guide the generation of candidate corrections.

Fine-tuning. GIDCL (Yan et al., 2024b) introduces a new pipeline: an implicit cleaning stage using a fine-tuned local LLM as the corrector, and an explicit cleaning stage that generates correction patterns. A ranking method is then employed to select the most suitable correction.

Pros & Cons. In data repair, prompt-based methods work well for generating simple repair rules and correcting small-scale value errors without extensive training data, but may yield inconsistent results for larger datasets that need consistent corrections. Fine-tuning learns stable repair patterns and ensures consistency across similar errors, but requires curated data and retraining.

3.4.3 Data Imputation

Data imputation focuses on inferring and filling in unobserved or missing elements. Prior non-LM approaches evolved from statistical methods to ML methods (Jäger et al., 2021a). Similar to the data repair, data imputation is also a generative task, and both LLM-centric methods and LM-in-the-loop methods are explored.

LM-centric. The method proposed in (Narayan et al., 2022) directly prompts the LLM with few-shot examples to perform imputation. Rather than relying on a single prompt and model, LLM-Forest (He et al., 2024) constructs a bipartite graph for each feature and integrates information from multiple LLM-based few-shot learning "trees" into a forest by merging these graphs. It then performs weighted voting to derive the final imputation result. Fine-tuning methods are also explored. The approach in (Ding et al., 2024b) applies LoRA to fine-tune the LLM, incorporates existing data as relevant knowledge into the prompt, and subsequently generates candidate values for imputation. Similarly, CRILM (Hayat and Hasan, 2024) fine-tunes an SLM using the outputs of an LLM, aiming to balance efficiency and accuracy.

LM-in-the-loop. Beyond relying solely on LLMs, LLM-in-the-loop methods have also been introduced. UnIMP (Wang et al., 2025a) first constructs cell-oriented hypergraphs by using the LLM as an encoder to generate feature representations. It then applies high-order message passing to aggregate and propagate information across the dataset, employing the LLM as a decoder to generate imputations based on the aggregated features.

Pros & Cons. LM-centric approaches can directly infer missing values but may overfit patterns or hallucinate plausible yet incorrect imputations. LLM-in-the-loop setups combine LLM reasoning with external rules or models, improving reliability and interpretability, though this adds orchestration overhead and requires careful design.

Summary and take-away. LMs, as generative models, are particularly well suited for data cleaning. This area has attracted substantial research attention and is progressing rapidly, with both LLM-centric and LM-in-the-loop approaches being actively explored.

3.5 Data Transformation

Touchpoint. In contrast to data-centric phases like data acquisition, data integration, and data cleaning, data transformation is more task-oriented and instruction-driven, aiming to bridge the gap between raw data and downstream tasks. While earlier studies leveraged SLMs for encoding-based transformation, recent advances increasingly turn to LLMs for their stronger instruction-following abilities which make them well-suited for instruction-driven data transformation, where

methods such as prompt engineering and LLM-as-encoder are leveraged to align data with specific requirements. In this paper, we survey formation transformation and semantic transformation.

3.5.1 Format Transformation

Format transformation aims to restructure records to conform to a specified target format schema. Early works address this using carefully designed operations (Li et al., 2023a; Chen et al., 2023).

Recently, LLMs are widely employed in a **Prompt-based** manner to automate format transformation. Researchers show that LLMs, like GPT-3.5 and GPT-4, can automatically recognize non-relational patterns and convert them into structured outputs with few-shot prompting (Huang and Wu, 2024). This idea is further expanded to multi-table scenarios, taking a higher-level overview zoom-in–zoom-out approach, showing a notable performance in the cross-table transformations (Huang et al., 2024). Instead of directly generating the transformed content, DataMorpher (Sharma et al., 2025) introduces a novel paradigm that first generates code and then executes it for data transformation. SQLMorpher (Sharma et al., 2023) proposes a new benchmark and applies similar code generation techniques, demonstrating the effectiveness of LLM-based multi-step transformations.

3.5.2 Semantic Transformation

Semantic transformation seeks to map tabular content into structured knowledge, such as functions, rules, or logical forms. Both the LM-as-encoder method and prompt engineering are explored.

LM-as-Encoder. Auto-Formula (Chen et al., 2024) adopts a retrieval-based approach, where tables are encoded using SentenceBERT (Reimers and Gurevych, 2019) to retrieve relevant spreadsheets and formulas. This retrieval augments the table-to-formula generation process by incorporating semantically similar examples.

Prompt Engineering. A benchmark is introduced in SPREADSHEETBENCH (Ma et al., 2024) that integrates user intents, tabular data, and example outputs, evaluating the performance of LLMs in generating executable code for complex spreadsheet manipulations using real-user prompts. In addition, TabAF (Wang et al., 2025c) employs a dual-prompting strategy to generate both answers and corresponding formulas for table-based question answering, leveraging perplexity-based selection to optimize the final output.

Pros & Cons. LM-as-Encoder approaches generate general-purpose representations that help align semantics, but they may miss fine-grained contextual cues. Prompt engineering provides flexible control over how semantic mappings are specified, yet its effectiveness can suffer if prompts fail to clearly define complex field dependencies or contextual constraints.

Summary and take-away. With the integration of advanced prompt engineering and LLM-as-encoder orchestration, LLMs present a novel tool for data transformation. LLM-based methods are capable of handling diverse data formats and complex user intents, thereby broadening the scope and scalability of instruction-driven transformation workflows.

3.6 LMs Applicable to Diverse Tasks

Alongside task-specific methods, unified data preprocessors are also being explored.

Prompt Engineering. A study in (Narayan et al., 2022) demonstrates that GPT-3-175B, the largest variant of GPT-3, achieves strong performance for tabular data preparation even under zero or few-shot prompting. A retrieval-augmented pipeline is introduced in UniDM (Qian et al., 2024). Cocoon (Zhang et al., 2024c) extends this insight by mimicking human cleaning and decomposing data cleaning into sub-tasks. The framework in (Zhang et al., 2024b) integrates prompt techniques like zero/few-shot, batch prompting, and feature selection for tabular data preparation.

Fine-Tuning-Based Strategies. Many work uses fine-tuning to encode knowledge of tabular data and preparation operations into LLMs. TableGPT (Li et al., 2023b) introduces table-tuning on 18 synthetic tasks, enabling strong table-centric reasoning, consistently outperforming GPT 3.5. Jellyfish (Zhang et al., 2024a) trains a local LLM on a small multi-task corpus using curated instructions and distilled reasoning from a larger model for data preprocessing. Unicorn (Tu et al., 2023) pioneers a unified model for seven data matching tasks (e.g., entity matching, schema matching). A SpreadsheetLLM is introduced in (Dong et al., 2024). MELD (Yan et al., 2024a) introduces a Mixture-of-Experts (MoE) method for low-resource data preprocessing with a router network that selects the top-k experts to handle queries.

3.7 Frameworks for Full Pipeline

Besides focusing on LMs, recent research has begun to explore how to orchestrate the entire pipeline by adopting multi-agent frameworks that allow multiple agents to work together with planning, reasoning, and execution. AutoPrep (Fan et al., 2024a) combines a Planner that generates logical workflows, Programmers that produce step-wise code, and an Executor that runs the code to handle question-aware tabular data preparation. Pipeline-Agent (Ge et al., 2025) formulates data preparation as an iterative decision-making process, where an agent selects operations, executes them, and adjusts subsequent steps based on intermediate results.

4 Challenges and Guidance

4.1 Challenges

Effectiveness and Robustness. A key challenge of LM-enabled methods is their effectiveness and robustness since LMs can produce unfaithful or logically inconsistent outputs (i.e., hallucinations). To mitigate this, recent works combine LMs with external retrieval or structured knowledge sources. For example, RAG-based pipelines provide factual grounding to reduce hallucinations, KG-RAG4SM (Ma et al., 2025) and KcMF (Xu et al., 2024) demonstrate how integrating knowledge graphs can verify or enrich model predictions, improving stability. However, scalable and generalizable solutions for robust, repeatable orchestration remain an open challenge.

Cost and Scalability. In real-world data pipelines that demand low latency and predictable resource budgets, balancing effectiveness with efficiency has become a practical priority. Recent research addressed this problem through various techniques. One common direction is to leverage traditional methods or SLMs to filter candidates and reduce the load on LLM components, as seen in frameworks like Magneto (Liu et al., 2024b), ReMatch (Sheetrit et al., 2024), and Prompt-Matcher (Feng et al., 2024). Another promising approach distills LLM knowledge into lighter SLMs, such as in Jellyfish (Zhang et al., 2024a) and CRILM (Hayat and Hasan, 2024), achieving comparable performance with low inference cost.

To operationalize the evaluation, we provide a discussion of benchmarks, metrics, and other evaluation aspects in Appendix D, as well as two tables in Appendix D.1 and D.2: a standardized reporting template and a robustness checklist.

4.2 Design Guidance

Drawing on the foregoing discussion, we present targeted recommendations for practitioners and a set of research directions in Appendix F.

5 Future Directions

5.1 Resource-Efficient LLM Pipelines

LLMs inevitably impose high computational and energy costs despite their strong capabilities (Hoffmann et al., 2022). Moreover, the input size of LLMs is inherently limited, making it difficult to encode the whole table within a single inference pass. At the same time, the volume of real-world data generated across various domains has grown at an unprecedented rate. While techniques for parameter-efficient fine-tuning and inference acceleration are proposed, the efficiency is still not competitive compared to traditional learning-based methods. Moreover, it remains challenging to balance accuracy and cost in high-throughput settings.

5.2 Interactions Across Different Phases

Most existing work treats tabular data preparation phases as independent modules. However, in practice, these phases are tightly coupled and can benefit from joint modeling. For example, cleaning and integration strongly influence each other: improved cleaning can reduce noise and inconsistencies, thus facilitating more accurate data integration. Conversely, effective integration can expose redundancies, conflicts, or outliers across sources, which in turn provides valuable context for more informed and targeted cleaning. Coordinating the entire process remains challenging due to the heterogeneous nature of phases, each of which may require distinct strategies. Moreover, the variability in downstream requirements further complicates the development of a one-size-fits-all solution.

6 Conclusions

LMs are reshaping tabular data preparation with flexible and effective solutions across acquisition, integration, cleaning, and transformation tasks. This survey offers a comprehensive review of why LMs are suitable to empower tabular data preparation and how LMs act as enablers for it, including LM-centric and LM-in-the-loop strategies. We examine a broad set of techniques, organize recent developments around shared paradigms, and summarize how existing methods are incorporated into practical workflows with future directions.

Limitations

This survey focuses primarily on tabular data preparation methods. However, in real-world scenarios, many other types of data, such as text, images, or time-series, also require specific preparation strategies. Our analysis does not comprehensively cover these diverse modalities, and thus some conclusions and techniques may not generalize beyond tabular data. For those interested in the preparation of other data types, Appendix G provides a brief overview of representative benchmarks and surveys. In addition, tabular data preparation is inherently connected to downstream tasks, including analytics, modeling, and decision-making processes. However, this survey explicitly limits its scope to the preparation of data itself, without a detailed discussion of these related downstream applications. The interaction and integration with these tasks remain an important area for future exploration.

References

Asma Ben Abacha, Wen-wai Yim, Yujuan Fu, Zhaoyi Sun, Meliha Yetisgen, Fei Xia, and Thomas Lin. 2024. MEDEC: A benchmark for medical error detection and correction in clinical notes. *CoRR*, abs/2412.19260.

Sarah Alnegheimish, Linh Nguyen, Laure Berti-Équille, and Kalyan Veeramachaneni. 2024. Can large language models be anomaly detectors for time series? In *11th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2024, San Diego, CA, USA, October 6-10, 2024*, pages 1–10. IEEE.

Anthropic. 2023. Anthropic claude. `https://www.anthropic.com/index/claude`.

Gustavo EAPA Batista and Maria Carolina Monard. 2003. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533.

Olivier Binette and Rebecca C. Steorts. 2020. (almost) all of entity resolution. *CoRR*, abs/2008.04443.

Jason Brownlee. 2020. *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery.

Raul Castro Fernandez, Ziawasch Abedjan, Famien Koko, Gina Yuan, Samuel Madden, and Michael Stonebraker. 2018. Aurum: A data discovery system. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 1001–1012.

Yaping Chai, Haoran Xie, and Si-Zhao Joe Qin. 2025. Text data augmentation for large language models: A comprehensive survey of methods, challenges, and opportunities. *CoRR*, abs/2501.18845.

Sibei Chen, Yeye He, Weiwei Cui, Ju Fan, Song Ge, Haidong Zhang, Dongmei Zhang, and Surajit Chaudhuri. 2024. Auto-formula: Recommend formulas in spreadsheets using contrastive learning for table representations. *Proceedings of the ACM on Management of Data*, 2(3):1–27.

Sibei Chen, Nan Tang, Ju Fan, Xuemi Yan, Chengliang Chai, Guoliang Li, and Xiaoyong Du. 2023. Haipipe: Combining human-generated and machine-generated pipelines for data preparation. *Proc. ACM Manag. Data*, 1(1):91:1–91:26.

Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D Davison. 2020. Table search using a deep contextualized language model. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 589–598.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. Bert: Pre-training of deep bidirectional transformers for language understanding. In *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)*, pages 4171–4186.

Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Anh Tuan Luu, and Shafiq Joty. 2024a. Data augmentation using llms: Data perspectives, learning paradigms and challenges. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 1679–1705. Association for Computational Linguistics.

Zhicheng Ding, Jiahao Tian, Zhenkai Wang, Jinman Zhao, and Siyang Li. 2024b. Semantic understanding and data imputation using large language model to accelerate recommendation system. *CoRR*, abs/2407.10078.

Haoyu Dong, Jianbo Zhao, Yuzhang Tian, Junyu Xiong, Shiyu Xia, Mengyu Zhou, Yun Lin, José Cambronero, Yeye He, Shi Han, et al. 2024. Spreadsheetllm: Encoding spreadsheets for large language models. *arXiv preprint arXiv:2407.09025*.

Yuyang Dong, Chuan Xiao, Takuma Nozawa, Masafumi Enomoto, and Masafumi Oyamada. 2023. Deepjoin: Joinable table discovery with pre-trained language models. *Proc. VLDB Endow.*, 16(10):2458–2470.

Grace Fan, Jin Wang, Yuliang Li, Dan Zhang, and Renée J. Miller. 2023. Semantics-aware dataset discovery from data lakes with contextualized column-based representation learning. *Proc. VLDB Endow.*, 16(7):1726–1739.

Meihao Fan, Ju Fan, Nan Tang, Lei Cao, Guoliang Li, and Xiaoyong Du. 2024a. Autoprep: Natural language question-aware data preparation with a multi-agent framework. *CoRR*, abs/2412.10422.

Meihao Fan, Xiaoyue Han, Ju Fan, Chengliang Chai, Nan Tang, Guoliang Li, and Xiaoyong Du. 2024b. Cost-effective in-context learning for entity resolution: A design space exploration. In *40th IEEE International Conference on Data Engineering, ICDE 2024, Utrecht, The Netherlands, May 13-16, 2024*, pages 3696–3709. IEEE.

Longyu Feng, Huahang Li, and Chen Jason Zhang. 2024. Cost-aware uncertainty reduction in schema matching with GPT-4: the prompt-matcher framework. *CoRR*, abs/2408.14507.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Yuhang Ge, Yachuan Liu, Yuren Mao, and Yunjun Gao. 2025. Text-to-pipeline: Bridging natural language and data preparation pipelines. *CoRR*, abs/2505.15874.

Chenjuan Guo, Cornelia Hedeler, Norman W. Paton, and Alvaro A. A. Fernandes. 2013. Matchbench: Benchmarking schema matching algorithms for schematic correspondences. In *Big Data - 29th British National Conference on Databases, BNCOD 2013, Oxford, UK, July 8-10, 2013. Proceedings*, volume 7968 of *Lecture Notes in Computer Science*, pages 92–106. Springer.

Yuxiang Guo, Zhonghao Hu, Yuren Mao, Baihua Zheng, Yunjun Gao, and Mingwei Zhou. 2025a. Birdie: Natural language-driven table discovery using differentiable search index. *arXiv preprint arXiv:2504.21282*.

Yuxiang Guo, Yuren Mao, Zhonghao Hu, Lu Chen, and Yunjun Gao. 2025b. Snoopy: Effective and efficient semantic join discovery via proxy columns. *IEEE Transactions on Knowledge and Data Engineering*.

Mazhar Hameed and Felix Naumann. 2020a. Data preparation: A survey of commercial tools. *SIGMOD Rec.*, 49(3):18–29.

Mazhar Hameed and Felix Naumann. 2020b. Data preparation: A survey of commercial tools. *ACM SIGMOD Record*, 49(3):18–29.

Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. 2002. Outlier detection using replicator neural networks. In *International conference on data warehousing and knowledge discovery*, pages 170–180. Springer.

Ahatsham Hayat and Mohammad Rashedul Hasan. 2024. CLAIM your data: Enhancing imputation accuracy with contextual large language models. *CoRR*, abs/2405.17712.

Xinrui He, Yikun Ban, Jiaru Zou, Tianxin Wei, Curtiss B. Cook, and Jingrui He. 2024. Llm-forest for health tabular data imputation. *CoRR*, abs/2410.21520.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. 2020. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Martin Eisenschlos. 2021. 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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 512–519. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Mueller, Francesco Piccinno, and Julian Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *CoRR*, abs/2203.15556.

Zezhou Huang, Jia Guo, and Eugene Wu. 2024. Transform table to database using large language models. In *Proceedings of Workshops at the 50th International Conference on Very Large Data Bases, VLDB 2024, Guangzhou, China, August 26-30, 2024*. VLDB.org.

Zezhou Huang and Eugene Wu. 2024. Relationalizing tables with large language models: The promise and challenges. In *40th International Conference on Data Engineering, ICDE 2024 - Workshops, Utrecht, Netherlands, May 13-16, 2024*, pages 305–309. IEEE.

Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. 2021a. A benchmark for data imputation methods. *Frontiers Big Data*, 4:693674.

Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. 2021b. A benchmark for data imputation methods. *Frontiers Big Data*, 4:693674.

Abhinav Jain, Hima Patel, Lokesh Nagalapatti, Nitin Gupta, Sameep Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Ruhi Sharma Mittal, and Vitobha Munigala. 2020. Overview and importance of data quality for machine learning tasks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3561–3562.

Mustafa Abdalrassual Jassim and Sarah N. Abdulwahid. 2021. Data mining preparation: Process, techniques and major issues in data analysis. *IOP Conference Series: Materials Science and Engineering*, 1090(1):012053.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Christos Koutras, George Siachamis, Andra Ionescu, Kyriakos Psarakis, Jerry Brons, Marios Fragkoulis, Christoph Lofi, Angela Bonifati, and Asterios Katsifodimos. 2021. Valentine: Evaluating matching techniques for dataset discovery. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 468–479. IEEE.

Christos Koutras, Jiani Zhang, Xiao Qin, Chuan Lei, Vasileios Ioannidis, Christos Faloutsos, George Karypis, and Asterios Katsifodimos. 2024. Omnimatch: Effective self-supervised any-join discovery in tabular data repositories. *CoRR*, abs/2403.07653.

Mark A Kramer. 1991. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243.

Sanjay Krishnan and Eugene Wu. 2019. Alphaclean: Automatic generation of data cleaning pipelines. *arXiv preprint arXiv:1904.11827*.

Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. 2023. Open-wikitable : Dataset for open domain question answering with complex reasoning over table. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 8285–8297. Association for Computational Linguistics.

Guoliang Li, Xuanhe Zhou, and Xinyang Zhao. 2024a. LLM for data management. *Proc. VLDB Endow.*, 17(12):4213–4216.

Huahang Li, Shuangyin Li, Fei Hao, Chen Jason Zhang, Yuanfeng Song, and Lei Chen. 2024b. Booster: Leveraging large language models for enhancing entity resolution. In *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, Singapore, May 13-17, 2024*, pages 1043–1046. ACM.

Peng Li, Yeye He, Cong Yan, Yue Wang, and Surajit Chaudhuri. 2023a. Auto-tables: Synthesizing multi-step transformations to relationalize tables without using examples. *Proc. VLDB Endow.*, 16(11):3391–3403.

Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. 2023b. Table-gpt: Table-tuned GPT for diverse table tasks. *CoRR*, abs/2310.09263.

Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *Proc. VLDB Endow.*, 14(1):50–60.

Yuliang Li, Xiaolan Wang, Zhengjie Miao, and Wang-Chiew Tan. 2021. Data augmentation for ml-driven data preparation and integration. *Proceedings of the VLDB Endowment*, 14(12):3182–3185.

Jun Liu, Chaoyun Zhang, Jiaxu Qian, Minghua Ma, Si Qin, Chetan Bansal, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. 2024a. Large language models can deliver accurate and interpretable time series anomaly detection.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. 2023. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-radiology*, 1(2):100017.

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

Yurong Liu, Eduardo Peña, Aécio S. R. Santos, Eden Wu, and Juliana Freire. 2024b. Magneto: Combining small and large language models for schema matching. *CoRR*, abs/2412.08194.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 11065–11082. Association for Computational Linguistics.

Weizheng Lu, Jing Zhang, Ju Fan, Zihao Fu, Yueguo Chen, and Xiaoyong Du. 2025. Large language model for table processing: A survey. *Frontiers of Computer Science*, 19(2):192350.

Chuangtao Ma, Sriom Chakrabarti, Arijit Khan, and Bálint Molnár. 2025. Knowledge graph-based retrieval-augmented generation for schema matching. *CoRR*, abs/2501.08686.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. 2024. Spreadsheetbench: Towards challenging real world spreadsheet manipulation. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Yury A. Malkov and Dmitry A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(4):824–836.

What Is Data Mining. 2006. Data mining: Concepts and techniques. *Morgan Kaufinann*, 10(559-569):4.

Omer Nahum, Nitay Calderon, Orgad Keller, Idan Szpektor, and Roi Reichart. 2024. Are llms better than reported? detecting label errors and mitigating their effect on model performance. *CoRR*, abs/2410.18889.

Avanika Narayan, Ines Chami, Laurel J. Orr, and Christopher Ré. 2022. Can foundation models wrangle your data? *Proc. VLDB Endow.*, 16(4):738–746.

Mona Nashaat, Aindrila Ghosh, James Miller, and Shaikh Quader. 2021. Tabreformer: Unsupervised representation learning for erroneous data detection. *ACM/IMS Transactions on Data Science*, 2(3):1–29.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *CoRR*, abs/2307.06435.

Marcello Nesca, Alan Katz, Carson K Leung, and Lisa M Lix. 2022. A scoping review of preprocessing methods for unstructured text data to assess data quality. *International Journal of Population Data Science*, 7(1):1757.

Wei Ni, Kaihang Zhang, Xiaoye Miao, Xiangyu Zhao, Yangyang Wu, Yaoshu Wang, and Jianwei Yin. 2025. Zeroed: Hybrid zero-shot error detection through large language model reasoning. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, pages 3126–3139. IEEE Computer Society.

Wei Ni, Kaihang Zhang, Xiaoye Miao, Xiangyu Zhao, Yangyang Wu, and Jianwei Yin. 2024. Iterclean: An iterative data cleaning framework with large language models. In *ACM Turing Award Celebration Conference 2024, ACM-TURC 2024, Changsha, China, July 5-7, 2024*. ACM.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Marcel Parciak, Brecht Vandevoort, Frank Neven, Liesbet M. Peeters, and Stijn Vansummeren. 2024. Schema matching with large language models: an experimental study. In *Proceedings of Workshops at the 50th International Conference on Very Large Data Bases, VLDB 2024, Guangzhou, China, August 26-30, 2024*. VLDB.org.

Ralph Peeters and Christian Bizer. 2021. Dual-objective fine-tuning of BERT for entity matching. *Proc. VLDB Endow.*, 14(10):1913–1921.

Ralph Peeters, Aaron Steiner, and Christian Bizer. 2025. Entity matching using large language models. In *Proceedings 28th International Conference on Extending Database Technology, EDBT 2025, Barcelona, Spain, March 25-28, 2025*, pages 529–541. OpenProceedings.org.

Danrui Qi and Jiannan Wang. 2024. Cleanagent: Automating data standardization with llm-based agents. *CoRR*, abs/2403.08291.

Yichen Qian, Yongyi He, Rong Zhu, Jintao Huang, Zhijian Ma, Haibin Wang, Yaohua Wang, Xiuyu Sun, Defu Lian, Bolin Ding, et al. 2024. Unidm: a unified framework for data manipulation with large language models. *Proceedings of Machine Learning and Systems*, 6:465–482.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Anish Das Sarma, Lujun Fang, Nitin Gupta, Alon Y. Halevy, Hongrae Lee, Fei Wu, Reynold Xin, and Cong Yu. 2012. Finding related tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 817–828. ACM.

Mayssam Sayyadian, Hieu LeKhac, AnHai Doan, and Luis Gravano. 2007. Efficient keyword search across heterogeneous relational databases. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 346–355. IEEE Computer Society.

Nabeel Seedat and Mihaela van der Schaar. 2024. Matchmaker: Self-improving large language model programs for schema matching. *CoRR*, abs/2410.24105.

Silvia Seoni, Alen Shahini, Kristen M. Meiburger, Francesco Marzola, Giulia Rotunno, U. Rajendra Acharya, Filippo Molinari, and Massimo Salvi. 2024. All you need is data preparation: A systematic review of image harmonization techniques in multi-center/device studies for medical support systems. *Comput. Methods Programs Biomed.*, 250:108200.

Ankita Sharma, Xuanmao Li, Hong Guan, Guoxin Sun, Liang Zhang, Lanjun Wang, Kesheng Wu, Lei Cao, Erkang Zhu, Alexander Sim, Teresa Wu, and Jia Zou. 2023. Automatic data transformation using large language model - an experimental study on building energy data. In *IEEE International Conference on Big Data, BigData 2023, Sorrento, Italy, December 15-18, 2023*, pages 1824–1834. IEEE.

Ankita Sharma, Jaykumar Tandel, Xuanmao Li, Lanjun Wang, Anna Fariha, Liang Zhang, Syed Arsalan Ahmed Naqvi, Irbaz Bin Riaz, Lei Cao, and Jia Zou. 2025. DATAMORPHER: Automatic Data Transformation using LLM-Based Zero-Shot Code Generation . In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, pages 4536–4539, Los Alamitos, CA, USA. IEEE Computer Society.

Eitam Sheetrit, Menachem Brief, Moshik Mishaeli, and Oren Elisha. 2024. Rematch: Retrieval enhanced schema matching with llms. *CoRR*, abs/2403.01567.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Aaron Steiner, Ralph Peeters, and Christian Bizer. 2024. Fine-tuning large language models for entity matching. *CoRR*, abs/2409.08185.

Stefan Stieglitz, Milad Mirbabaie, Björn Ross, and Christoph Neuberger. 2018. Social media analytics – challenges in topic discovery, data collection, and data preparation. *International Journal of Information Management*, 39:156–168.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Jianhong Tu, Ju Fan, Nan Tang, Peng Wang, Guoliang Li, Xiaoyong Du, Xiaofeng Jia, and Song Gao. 2023. Unicorn: A unified multi-tasking model for supporting matching tasks in data integration. *Proc. ACM Manag. Data*, 1(1):84:1–84:26.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro Szekely. 2021. Retrieving complex tables with multi-granular graph representation learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1472–1482.

Jianwei Wang, Kai Wang, Ying Zhang, Wenjie Zhang, Xiwei Xu, and Xuemin Lin. 2025a. On llm-enhanced mixed-type data imputation with high-order message passing. *CoRR*, abs/2501.02191.

Jianwei Wang, Ying Zhang, Kai Wang, Xuemin Lin, and Wenjie Zhang. 2024a. Missing data imputation with uncertainty-driven network. *Proceedings of the ACM on Management of Data*, 2(3):1–25.

Qiming Wang and Raul Castro Fernandez. 2023. Solo: Data discovery using natural language questions via A self-supervised approach. *Proc. ACM Manag. Data*, 1(4):262:1–262:27.

Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xuanang Chen, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. 2025b. Match, compare, or select? an investigation of large language models for entity matching. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 96–109. Association for Computational Linguistics.

Tianshu Wang, Hongyu Lin, Xiaoyang Chen, Xianpei Han, Hao Wang, Zhenyu Zeng, and Le Sun. 2024b. Match, compare, or select? an investigation of large language models for entity matching. *CoRR*, abs/2405.16884.

Zhongyuan Wang, Richong Zhang, and Zhijie Nie. 2025c. General table question answering via answer-formula joint generation. *CoRR*, abs/2503.12345.

Shiwen Wu, Qiyu Wu, Honghua Dong, Wen Hua, and Xiaofang Zhou. 2023. Blocker and matcher can mutually benefit: A co-learning framework for low-resource entity resolution. *Proceedings of the VLDB Endowment*, 17(3):292–304.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, and Tao Gui. 2023. The rise and potential of large language model based agents: A survey. *CoRR*, abs/2309.07864.

Yongqin Xu, Huan Li, Ke Chen, and Lidan Shou. 2024. Kcmf: A knowledge-compliant framework for schema and entity matching with fine-tuning-free llms. *CoRR*, abs/2410.12480.

Mengyi Yan, Yaoshu Wang, Kehan Pang, Min Xie, and Jianxin Li. 2024a. Efficient mixture of experts based on large language models for low-resource data preprocessing. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 3690–3701. ACM.

Mengyi Yan, Yaoshu Wang, Yue Wang, Xiaoye Miao, and Jianxin Li. 2024b. GIDCL: A graph-enhanced interpretable data cleaning framework with large language models. *Proc. ACM Manag. Data*, 2(6):236:1–236:29.

Yisehac Yohannes and John F Hoddinott. 1999. Classification and regression trees: an introduction.

Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. 2024a. Jellyfish: Instruction-tuning local large language models for data preprocessing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 8754–8782. Association for Computational Linguistics.

Haochen Zhang, Yuyang Dong, Chuan Xiao, and Masafumi Oyamada. 2024b. Large language models as data preprocessors. In *Proceedings of Workshops at the 50th International Conference on Very Large Data Bases, VLDB 2024, Guangzhou, China, August 26-30, 2024*. VLDB.org.

Shichao Zhang, Chengqi Zhang, and Qiang Yang. 2003. Data preparation for data mining. *Applied artificial intelligence*, 17(5-6):375–381.

Shuo Zhang, Zezhou Huang, and Eugene Wu. 2024c. Data cleaning using large language models. *CoRR*, abs/2410.15547.

Ting Zhang, Xin Huang, Wen Zhao, Shaohuang Bian, and Peng Du. 2023. Logprompt: A log-based anomaly detection framework using prompts. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*, pages 1–8. IEEE.

Erkang Zhu, Fatemeh Nargesian, Ken Q. Pu, and Renée J. Miller. 2016. LSH ensemble: Internet-scale domain search. *Proc. VLDB Endow.*, 9(12):1185–1196.

Jingyu Zhu, Xintong Zhao, Yu Sun, Shaoxu Song, and Xiaojie Yuan. 2024. Relational data cleaning meets artificial intelligence: A survey. *Data Science and Engineering*, pages 1–28.

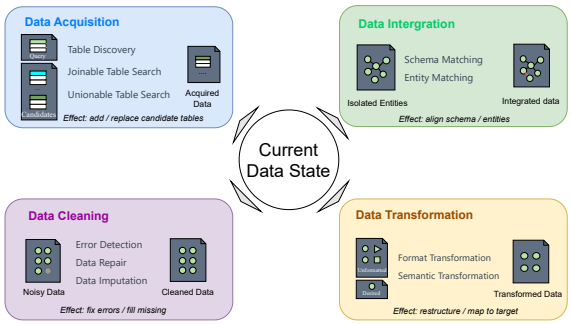

Figure 3: An LM-empowered tabular data-preparation pipeline covering data collection, integration, cleaning, and transformation tasks.

A Language Models

Language models (LMs), often transformer-based (Vaswani et al., 2017), are trained on large text datasets to understand and generate human language. They operate auto-regressively, predicting the next word based on prior context as follows:

$$\theta_{LM} = \arg\max_{\theta} \sum_{i} \log P(d_i | d_{i-1}, \cdots; \theta) \quad (1)$$

Where d_i is a text data token and θ_{LM} is the parameter of LM. In the early days, LMs were primarily small language models (SLMs), such as T5 (Raffel et al., 2020) and Bert (Devlin et al., 2019b). Recently, driven by the scaling law (Kaplan et al., 2020; Henighan et al., 2020), which suggests that larger models tend to perform better, LLMs, such as GPT-4 (OpenAI, 2023), LLaMA (Touvron et al., 2023) and Claude (Anthropic, 2023), are emerging.

B Task Definitions

The formal task definitions that follow correspond to the processing pipeline illustrated in Figure 3. These definitions aim to provide an operational, reproducible taxonomy that links the conceptual pipeline in the main text to measurable experimental tasks.

B.1 Data Acquisition

Definition 1 *(Table Discovery):* Given a natural language query q and a table repository $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$, the goal of table discovery is to retrieve the top-k tables $T_t \in \mathcal{T}$ that are most relevant to the query. The relevance is measured by a ranking score $R_{\text{disc}}(q, T_t) = \text{Rel}(q, T_t)$, where Rel captures the semantic alignment between the

query and the content (e.g., captions, schemas, and cell values) of the table.

Definition 2 *(Joinable table search):* Given a query table T_q with a specified column C_q, and a table collection $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$, the goal of joinable table search is to retrieve the top-k tables $T_t \in \mathcal{T}$ such that there exists a column $C_t \in T_t$ maximizing a joinability score: $R_{\text{join}}(C_q, C_t) = \text{JoinSim}(C_q, C_t)$, where JoinSim measures the likelihood that C_q and C_t can serve as join keys.

Definition 3 *(Unionable Table Search):* Given a query table T_q^U and a table collection $\mathcal{T}$, top-k table union search retrieves a subset $\mathcal{T}_q \subset \mathcal{T}$, such that $|\mathcal{T}_q| = k$, and for any $T \in \mathcal{T}_q$, $T' \in \mathcal{T} \setminus \mathcal{T}_q$, it holds that: $R(T_q^U, T) > R(T_q^U, T')$, where $R(T_q^U, T_t)$ denotes a table-level unionability score.

B.2 Data Integration

Definition 4 *(Schema Matching):* Let S_s and S_t denote the source and target schemas. Each schema S comprises a set of tables $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$, where each table T_i contains attributes $\mathcal{A}_i = \{A_{i1}, A_{i2}, \ldots, A_{ik}\}$. Schema mapping aims to learn a function $f : \mathcal{A}_s \to \mathcal{A}_t \cup \{\varnothing\}$ that aligns each source attribute from the source schema S_s with a target attribute in the target schema S_t or assigns it to $\varnothing$ if unmatched.

Definition 5 *(Entity Matching):* Given two sets of records $\mathcal{R}_s = \{r_1, r_2, \ldots, r_m\}$ and $\mathcal{R}_t = \{r'_1, r'_2, \ldots, r'_n\}$, entity matching aims to learn a function $g : \mathcal{R}_s \times \mathcal{R}_t \to \{0, 1\}$, where $g(r_i, r'_j) = 1$ indicates that r_i and r'_j refer to the same entity, while $g(r_i, r'_j) = 0$ indicates they do not match.

B.3 Data Cleaning

Definition 6 *(Error Detection):* Error detection aims to identify $\mathcal{E} = \{t_{ij} \in T \mid t_{ij} \text{ is erroneous}\}$ given the raw dataset T, where the determination of an "erroneous" cell can be based on various criteria such as violation of data integrity constraints, semantic inconsistencies, or statistical anomalies.

Definition 7 *(Data Repair):* Given a raw dataset T and a set of detected erroneous cells $\mathcal{E}$ identified through the error detection, data repair aims to construct a new dataset T' such that all cells $t_{ij} \in \mathcal{E}$ are rectified and T' contains no erroneous values.

Definition 8 *(Data Imputation):* Data imputation aims to impute the unobserved elements in the raw tabular dataset T, i.e., $m_{ij} = 0$ in the mask matrix M, and make the imputed dataset as close to the real complete dataset (if it exists) as possible.

B.4 Data Transformation

Definition 9 *(Format Transformation):* Given a dataset D with records $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ and a target format schema F, format transformation learns a function $\phi : \mathcal{R} \to \mathcal{R}^\star$ that restructures each r_i into $r_i^\star$ to conform to F.

Definition 10 *(Semantic Transformation):* Given a dataset D with records $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$, semantic transformation aims to learn a function $\psi : \mathcal{R} \to \mathcal{K}$ that maps tabular content to structured knowledge such as functions, rules, or logic forms.

C Taxonomy Table

A taxonomy table, containing information about data preparation phases, specific tasks in each phase, related models, and relevant framework and highlights is given in Table 1.

D Evaluation, Reporting Template and Checklist

Systematic evaluation starts with reliable benchmarks and well-defined metrics.

Benchmarks. While existing benchmarks for specific tasks are widely used, such as Matchbench (Guo et al., 2013)for schema matching, Imputationbench (Jäger et al., 2021b) for missing data imputation and Medec (Abacha et al., 2024) for error detection, they mostly target individual subtasks rather than covering the entire tabular data preparation pipeline, there is still a lack of unified benchmarks that capture the whole workflows.

Metrics. The evaluation involves two main categories of metrics: (1) evaluation on data preparation accuracy, and (2) evaluation on downstream task performance. The first category measures the quality of prepared data directly against a ground-truth version, using standard scores like Accuracy, F1-score and Root Mean Squared Error (RMSE) (Fan et al., 2024b; Nashaat et al., 2021; Li et al., 2020). The second category assesses practical utility by measuring the impact on a downstream task, such as comparing a model's

Phase	Task	Model	Framework	Highlight	Effectiveness	Cost
Data Discovery	*Table Discovery*	OpenWikiTable (Kweon et al., 2023)	LM-as-Encoder	Several encoder options (BERT and TAPAS)	★	★★
		Solo (Wang and Fernandez, 2023)	LM-as-Encoder	Triplet-level cell–attribute embeddings with ANN index	★★	★★★
		Hybrid-BERT (Chen et al., 2020)	Fine-Tuning & LM-as-Encoder	Fine-tuned BERT reranker with content selector	★★	★
		GTR (Wang et al., 2021)	LM-as-Encoder	Graph-based joint query–table matching	★★	★
		Birdie (Guo et al., 2025a)	Fine-Tuning & LM-as-Encoder & LM-as-Decoder	Differentiable search index learned end-to-end	★★★	★
	Joinable Table Search	Snoopy (Guo et al., 2025b)	LM-as-Encoder	Proxy-column embeddings via contrastive learning	★★	★★★
		DeepJoin (Dong et al., 2023)	Fine-Tuning & LM-as-Encoder	Fine-tuned column encoder with HNSW retrieval	★★	★★★
	Unionable Table Search	Starmie (Fan et al., 2023)	LM-as-Encoder	Table-level embeddings with bipartite matching rerank	★★	★★★
Data Integration	*Schema Mapping*	ReMatch (Sheetrit et al., 2024)	Prompt Engineering	Retrieval-augmented framework	★	★
		Matchmaker (Seedat and van der Schaar, 2024)	Prompt Engineering	Automated self-improving	★★	★
		Prompt-Matcher (Feng et al., 2024)	Prompt Engineering	Uncertainty reduction pipeline	★	★★★
		Magneto (Liu et al., 2024b)	Prompt Engineering	Combining SLM with LLM	★★	★★
		KcMF (Xu et al., 2024)	Prompt Engineering	Retrieval-based domain knowledge & Pseudo-code task instruction	★★★	★
		KG-RAG4SM (Ma et al., 2025)	Fine-Tuning & LM-as-Encoder	Retrieve large external knowledge graph	★★★	★
	Entity Linkage	MatchGPT(Peeters et al., 2025)	Prompt Engineering	Demonstrated the superiority of prompt-based approaches	★★	★
		ComEM (Wang et al., 2024b)	Prompt Engineering	Hybrid "match-compare-select" strategy	★★	★★
		BoostER (Li et al., 2024b)	Prompt Engineering	Uncertainty reduction pipeline	★	★★★
		BATCHER (Fan et al., 2024b)	Prompt Engineering	Question batching & Batched prompting	★	★★★
		(Steiner et al., 2024)	Fine-Tuning	Domain-specific fine-tuning	★★	★
		JointBERT (Peeters and Bizer, 2021)	Fine-Tuning	Fine-tuned on different tasks	★★	★
		Ditto (Li et al., 2020)	Fine-Tuning	Targeted textual augmentations & Domain-specific knowledge injection	★★	★★
Data Cleaning	*Data Imputation*	LLM-Imputer(Ding et al., 2024b)	Fine-Tuning & Prompt Engineering	Combining Fine-Tuning and Prompt Engineering	★	★
		CLAIM (Hayat and Hasan, 2024)	Fine-Tuning & Prompt Engineering	Generating natural language descriptors for missing values	★★	★★
		LLM-Forest (He et al., 2024)	Prompt Engineering & LM-as-Encoder	Graph-based neighbor identification + LLM forests	★★	★
		UnIMP (Wang et al., 2025a)	Prompt Engineering & LM-as-Encoder	Combining hypergraph message passing with LLM embeddings	★★★	★★
	Error Detection	LLM-as-a-Judge (Nahum et al., 2024)	Prompt Engineering	Ensembling LLMs to detect label errors	★★★	★
		SIGLLM(Alnegheimish et al., 2024)	Prompt Engineering	Role-play prompt-driven	★	★★★
		LLMAD(Liu et al., 2024a)	Prompt Engineering	Chain-of-thought prompt-enhanced	★★	★★
		LogPrompt(Zhang et al., 2023)	Prompt Engineering	Chain-of-thought reasoning	★★★	★★★
	Data Repair	GIDCL(Yan et al., 2024b)	Prompt Engineering	Few-shot inference of repair patterns with hypergraph modeling	★★	★★
		IterClean(Ni et al., 2024)	Prompt Engineering	Iterative generate-validate-refine loop	★★★	★★
Data Transformation	*Format Transformation*	CleanAgent (Qi and Wang, 2024)	Prompt Engineering & LM-as-Encoder	Automated data standardization via multi-agent LLM collaboration	★★	★★
		DataMorpher (Sharma et al., 2025)	Prompt Engineering & LM-as-Encoder	Automated Python code generation	★★★	★
		SQLMorpher (Sharma et al., 2023)	Prompt Engineering & LM-as-Encoder	Automated SQL code generation	★★	★★
		Tab2DB (Huang et al., 2024)	Prompt Engineering	"zoom-in"–"zoom-out" approach	★★	★
	Content Transformation	DataMorpher (Sharma et al., 2025)	Prompt Engineering	Automated Python code generation	★★★	★
		SQLMorpher (Sharma et al., 2023)	Prompt Engineering	Automated SQL code generation	★★	★★
General Model	*Integrated Tasks*	Cocoon (Zhang et al., 2024c)	Prompt Engineering	Decomposing data cleaning tasks and mimicking human cleaning processes	★★★	★★★
		Table-GPT (Li et al., 2023b)	Fine-Tuning	First LLM fine-tuned on diverse table tasks	★★	★★
		LLM Preprocessor (Zhang et al., 2024b)	Prompt Engineering	Integrating a series of SOTA prompt engineering techniques	★★★	★★
		Jellyfish (Zhang et al., 2024a)	Prompt Engineering	Knowledge Distillation	★★★	★★
		Unicorn (Tu et al., 2023)	Fine-Tuning	Generic Encoder + Mixture of LLM Experts	★★★	★
		MELD (Yan et al., 2024a)	Fine-Tuning	RAG + meta-path search + Mixture of LLM Experts	★★★	★
		Foundation model(Narayan et al., 2022)	Prompt Engineering	Processing structured data as text and executes tasks	★★	★

Table 1: A taxonomy of LM-enabled tabular data preparation methods, grouped by phase and specific task. For the 'Effectiveness' and 'Cost' columns, more stars indicate better performance and lower cost, respectively.

performance when trained on the original versus the prepared data (Hayat and Hasan, 2024; Fan et al., 2024a; Qi and Wang, 2024). Detailed definitions of these metrics are in the Appendix E.

This survey motivates the development of a benchmark paper that provides a unified framework for evaluating LM-assisted tabular data preparation. The benchmark is intended to address three key questions: (i) When do LM operators outperform rule-based or learned baselines across datasets? (ii) Which high-level design choices influence preparation quality, latency, and cost most? and (iii) How robust are pipelines across dataset characteristics and distribution?

D.1 Reporting Template

A reporting template, designed to standardize the description of experimental results such as latency, throughput, token budgets, retry policy, and seeds/configs, is given in Table 2.

D.2 Robustness Checklist

A robustness checklist, providing structured guidance for evaluating end-to-end pipeline robustness across multiple dimensions (e.g., multi-seed variance, decoding sensitivity, distribution shift), is given in Table 3.

E Metrics Definitions

Definition 11 *(Accuracy):* For a given dataset of n instances $\{(x_i, y_i)\}_{i=1}^{n}$, where y_i is the true label for input x_i, let $\hat{y}_i$ be the predicted label. Accuracy is the fraction of predictions that are correct. Using an indicator function $\mathbb{I}(\cdot)$, it is formally defined as:

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(\hat{y}_i = y_i)$$

Definition 12 *(F1-score):* In a binary classification context, let TP, FP, and FN be the counts of True Positives, False Positives, and False Negatives. Let:

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \quad , \quad \text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

The F1-score is calculated as:

$$\text{F1-score} = 2\cdot\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}} = \frac{2\text{TP}}{2\text{TP}+\text{FP}+\text{FN}}$$

Definition 13 *(Root Mean Squared Error - RMSE):* For a numerical prediction task, let $Y = \{y_1, \ldots, y_n\}$ be the set of n true values and $\hat{Y} = \{\hat{y}_1, \ldots, \hat{y}_n\}$ be the set of predicted values. RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Definition 14 *(Downstream Task Performance):* Let $\mathcal{M}_{\text{down}}$ be a downstream model (e.g., a classifier) and $\text{Metric}_{\text{down}}$ be its evaluation metric (e.g., Accuracy). The performance is determined by training $\mathcal{M}_{\text{down}}$ on $\mathcal{D}_{\text{prep}}$ and original raw data $\mathcal{D}_{\text{ori}}$. The performance of data preparation is the performance gap between prepared data and original data. The performance is computed as:

$$\begin{aligned}\text{Performance} = {} & \text{Metric}_{\text{down}}\big(\mathcal{M}_{\text{down}}(\mathcal{D}_{\text{prep}})\big) \\ & - \text{Metric}_{\text{down}}\big(\mathcal{M}_{\text{down}}(\mathcal{D}_{\text{ori}})\big)\end{aligned}$$

F Decision Flow & Design heuristics.

In this section, we offer a short decision flow to choose an LM role for a given data-preparation operator. We suggest starting by selecting the operator (acquisition, integration, cleaning, transformation), then pick the LM role that matches the operator's retrieval, precision, and cost needs. For acquisition, we recommend using an SLM as encoder with a retrieval pipeline and calling an LLM for reranking only when finer semantic discrimination is needed. For integration, a good approach is to use SLM or heuristic pre-filtering followed by selective LLM reranking/matching. For cleaning and transformation, we generally prefer LM-centric prompting or fine-tuning and LM-in-the-loop (encoder features with retrieval/verification) may be more suitable when stability or verifiability is required.

When dealing with tight latency or cost constraints, we suggest following an SLM-first rule: filter data at scale using SLMs or rules, and reserve LLM calls for smaller candidate sets. Where possible, consider distilling LLM behavior into SLMs to reduce inference costs. Additionally, incorporating retrieval-augmented verification or knowledge-graph checks can help mitigate the risk of hallucinations. It's also important to respect LLM input-length limits by indexing or segmenting larger tables. Lastly, we suggest reporting not just accuracy,

Model Name		
Pipeline Architecture		
Dataset	**Dataset Name**	
	Split Identifier	
Reported Metrics	**Accuracy**	
	F1	
	Precision	
	RMSE	
	MAE	
Wall-Time	**p50**	
	p95	
	Total Wall-time	
Throughput (items/sec)		
Input & Output Tokens		
Number of Retries		
Retry Policy		
Random Seeds		
Nondeterministic Params		
Brief Preprocessing Notes		

Table 2: Reporting Template for Experimental Results

Seed		
Nondeterministic Record		
Temperature		
Sampling Sensitivity		
Prompt-Paraphrase Robustness		
Judge-Ensemble & Verification Protocol	**Verification Setup**	
	Voting/Threshold Rules	
	Accept/Reject Rates	
	Model/Manual Review Config	
Dataset Variation	**Small Dataset**	
	Medium Dataset	
	Large Dataset	
Distribution-Shift	**No Shift**	
	Label Shift	
	Domain Shift	
Monitoring & Throughput Indicators	**p50**	
	p95	
	Error Rates	
	Alerting Thresholds	

Table 3: Robustness Checklist for Practitioner Guidance

but also cost, latency, and throughput metrics to facilitate a fair comparison.

G Data Preparation for Other Data Modalities

For those interested in the preparation of other data types, a brief overview of representative benchmarks and surveys is provided below.

For image data, (Seoni et al., 2024) offer a systematic review of image harmonization techniques in multi-center and multi-device studies, which is a critical component of data preparation. Their work analyzes common strategies such as grayscale normalization, color normalization, resampling, and denoising, and evaluates their impact on the performance of downstream models. In the domain of text data, the field is covered from multiple perspectives. (Nesca et al., 2022) provide a scoping review of traditional preprocessing methods for unstructured text, including techniques like stop word removal, tokenization, and punctuation removal, with a focus on assessing data quality. Addressing more recent advancements, (Chai et al., 2025) present a comprehensive survey on text data augmentation using LLMs, classifying modern techniques into categories such as Prompt-based, Retrieval-based, and Hybrid Augmentation. In contrast, our search did not identify recent, comprehensive surveys or benchmark papers specifically dedicated to the end-to-end data preparation pipelines for time series data and graph data. This suggests a potential research gap and highlights a valuable opportunity for future work to establish systematic reviews and benchmarks for these increasingly critical data modalities.