A Survey on LLMs for Spreadsheet Intelligence

Tuan-Quang Vuong, **Karim Tit**, **Maxime Cordy**
Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg
{quang.vuong, karim.tit, maxime.cordy}@uni.lu

Abstract

Spreadsheets play a critical role in modern data work and are one of the most popular tools for data analysis across various domains. Despite their prevalence, spreadsheet tasks often require writing complex formulas, cleaning tabular data, and a thorough understanding of the heterogeneous structure, which are error-prone and time-consuming, even for expert users. The integration of Large Language Models (LLMs) in spreadsheet environments represents a significant paradigm shift, moving data analysis and manipulation from manual formulas to natural language interaction. This survey reviews the emerging landscape of LLM applications for spreadsheet tasks, identifies key methodologies, core capabilities, available benchmarks, and persistent challenges. First, we formulate the spreadsheet intelligence problem as a workflow of independent stages. Next, we categorize existing research works based on the defined stages and further segregate them into a taxonomy of tasks. Moreover, we list downstream tasks and their corresponding benchmarks, forming an end-to-end pipeline. Finally, we discuss open challenges and outline future research directions towards trustworthy LLM systems in spreadsheet environments.

1 Introduction

Spreadsheet technologies are among the most ubiquitous data tools, used by practitioners for data-related tasks spanning from basic data recording to complex modeling and advanced computation. As such, they are a central component of digital productivity, with applications ranging from finance and data science to engineering and administration.

However, many practitioners rely on self-taught or informal expertise rather than formal instruction [Birch *et al.*, 2017], leading to the underutilization of advanced functionalities such as nested formulas, multi-step transformations, or macros. Moreover, typical spreadsheet workflows frequently involve repetitive operations that are time-consuming and error-prone, especially when performed manually. Traditional rule-based, hard-coded tools (e.g., in Visual Basic) partially solve these issues but lack the flexibility to accommodate the diversity and ambiguity of real-world spreadsheet workflows. Ultimately, the current state of practice underscores the need for intelligent spreadsheet systems that can interpret user intent, generalize patterns, and automate a variety of complex tasks [Hermans *et al.*, 2012].

Large language models (LLMs), leveraging their state-of-the-art capabilities in natural language processing and code generation, are seen as valuable candidates to bridge this gap by directly translating human intentions into spreadsheet actions. Indeed, several LLMs have demonstrated remarkable capabilities in understanding and reasoning to generate natural language, code files, and structured data [Jaimovitch-López *et al.*, 2022]. The integration of Copilot in Microsoft Excel and Gemini in Google Sheets marked a breakthrough in the application of generative artificial intelligence to daily data tasks, changing how users interact with spreadsheets.

Yet, the literature shows evidence that achieving automated spreadsheet intelligence with LLMs is not straightforward, as these models reveal limitations when solving related tasks, e.g., identifying tables or interpreting formatting information [Dong *et al.*, 2024]. Though these gaps clearly form a research challenge, the related literature is scattered, often focused on specific downstream tasks and making evaluations on fragmented benchmarks. With this paper, we bring structure to the research field of LLM-based spreadsheet intelligence and review the current state of the art to identify open challenges and support future research progressing in this field. Specifically, our contributions are as follows.

- We propose a unified problem formulation that organizes fragmented tasks into a consistent and generic flow. This enables the structuring of existing and future literature and provides a clear assessment of the remaining gaps.
- Within this workflow, we categorize existing works into a taxonomy of tasks, covering both pre-processing tasks before the LLM is prompted and downstream tasks.
- We identify and review an extensive range of models, datasets, and methods for solving these tasks. By surveying available benchmarks, we encourage consistent experimental protocols and reproducible research.
- We scope and discuss the current state of the art, identify

Preprint. Under review.

lessons learned, and highlight open challenges that remain unaddressed. Thereby, we reveal relevant research directions with the potential to significantly enhance the practical ability of LLMs to understand spreadsheets.

Overall, through this extensive survey, we establish foundations to aid future work, aiming for comprehensive, comparable, and scalable spreadsheet intelligence systems.

2 Problem Formulation and Methodology

Problem Formulation

Numerous works utilize LLMs for spreadsheet-related tasks. However, most of them tackle only one aspect of the problem, lacking a global view. We define **Spreadsheet Intelligence** as the overarching goal consisting of three main stages: Spreadsheet Decomposition, Pre-tokenization, and Downstream Tasks. We structure the study according to these stages, allowing researchers to customize and adapt the components independently to their architecture.

First, **Spreadsheet Decomposition** is the problem of detecting structured tables, generated charts, formatting information, and notes in a spreadsheet. A spreadsheet file may contain multiple sheets, each with various components in different formats. Feeding the entire file as input hurts overall performance. Hence, it is crucial to decompose these elements and process each one correctly. In Section 3, we discuss element detection methods from structured format (.xlsx, .csv) and unstructured PDFs. As tables generally encode more valuable information and charts can serve as direct input, researchers primarily study the table detection problem.

Next, **Pre-tokenization** denotes the mapping of decomposed elements into a structured representation suitable for LLMs/VLMs. Here, these heterogeneous components are transformed into consistent, tokenizable units, ranging from serialized, flattened table segments to chart descriptions or rendered images of tables. Section 4 surveys research on text and image modalities and mentions context enrichment methods to optimize and augment the input prompt. We also mention more advanced techniques for interacting with spreadsheets, including restructuring tables into a relational form for lossless retrieval and equipping agentic systems with external tools for seamless interaction and integration.

Lastly, applying LLMs to spreadsheet workflows ultimately leads to **Downstream Tasks & Benchmarks**. Section 5 lists spreadsheet intelligence tasks, along with available benchmarks. From our observation, it is necessary to curate more unified benchmarks that tackle a wider range of tasks.

Survey Methodology

This survey aims to provide a holistic examination of the current landscape of spreadsheet intelligence. To the best of our knowledge, this is the first work to consolidate and analyze research papers regarding LLMs usage specifically for spreadsheet data. Although surveys on handling general tabular data with LLMs exist (e.g., [Fang *et al.*, 2024]), they primarily focus on serializing tables to text or images. Thus, when applying LLMs to spreadsheet data, these methods overlook other informative representations, such as charts or formatting details, which encode valuable insights that may assist the reasoning process. This poses new challenges in properly encoding this information for LLMs. Furthermore, table layouts in spreadsheets are considered semi-structured, suggesting the need for an oracle module.

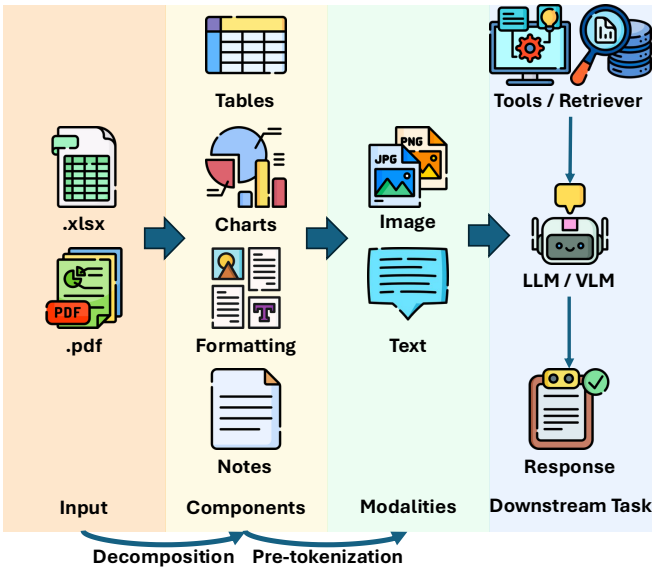

Figure 1: Workflow of LLM for Spreadsheet Intelligence

We gather 86 papers by systematically selecting works from multiple corpora: Google Scholar, ACL Anthology, IEEE Xplore, ACM Digital Library, and Arxiv, filtering for keywords: *LLM; Foundation Model; spreadsheet/tabular/table data*. We first screen papers based on their titles to filter out irrelevant studies, followed by an abstract-based screening to exclude papers that do not explicitly propose novel contributions. We also exclude studies that solely focus on simple and single tabular reasoning, resulting in 16 works. To expand the literature, we adopt the snowball approach by identifying articles that the selected studies cited or that cite them, and we apply the screening procedure again.

All in all, we review 50 publications. From this review, we formalize a workflow for spreadsheet intelligence that we illustrate in Figure 1. Then, we categorize different works into a taxonomy of tasks, summarized in Table 1.

3 Spreadsheet Decomposition

Spreadsheets are semi-structured data, meaning that while they exhibit specific structures, they often have inconsistent and irregular schemas. In simple scenarios, they can consist of a single table with regular rows and columns, consistent headers, and uniform data types. In many practical settings, however, they contain text, numbers, merged cells with inconsistent layouts, or even multiple tables and free-form text notes using unpredictable formatting. In extreme cases, a sheet can consist only of random blocks of text with no precise row-column semantics, and its layout is not readily interpretable without domain knowledge. Additionally, the presence of embedded charts and images further increases their structural complexity and complicates automated interpretation. In some cases, mapping the spreadsheet content to a

Stage	Task	Corresponding Papers
Spreadsheet Decomposition	Structured format	[Chen and Cafarella, 2014; Dou *et al.*, 2018; Fang *et al.*, 2021; Koci *et al.*, 2019]
	Unstructured format	[Hermans and Murphy-Hill, 2015; Li *et al.*, 2025; Shigarov *et al.*, 2016]
	Formatting Elements	[Chen *et al.*, 2024; Chen *et al.*, 2025; Dong *et al.*, 2024; Li *et al.*, 2023]
Pre-tokenization	Text-based Rep.	[Chen, 2023; Dong *et al.*, 2024; Fang *et al.*, 2024; Singha *et al.*, 2023; Sui *et al.*, 2024; Wang *et al.*, 2024a; Wang *et al.*, 2024b; Yu *et al.*, 2023]
	Image-based Rep.	[Borisova *et al.*, 2025; Deng *et al.*, 2024; Xia *et al.*, 2024]
	Prompt Optimization	[Chen *et al.*, 2019; Chen, 2023; Cheng *et al.*, 2022a; Cheng *et al.*, 2022b; Dong *et al.*, 2024; Fang *et al.*, 2024; Fu *et al.*, 2025]
	RAG	[Shi *et al.*, 2021; Sundar and Heck, 2023; Zhao *et al.*, 2023; Zhou *et al.*, 2025]
	Spreadsheet-to-SQL	[Jiang *et al.*, 2023; Li *et al.*, 2024]
	Agentic Workflow	[Chen *et al.*, 2025; Li *et al.*, 2023; Sui *et al.*, 2024; Yao *et al.*, 2023; Ye *et al.*, 2023]
Downstream Tasks	Question Answering	[Cheng *et al.*, 2022b; Dong *et al.*, 2024; Foroutan *et al.*, 2025; Jing *et al.*, 2025]
	Fact Verification	[Aly *et al.*, 2021; Thorne *et al.*, 2018]
	Editing & Handling	[Chen *et al.*, 2025; Indika and Molybog, 2025; Li *et al.*, 2023]
	Data Analysis	[He *et al.*, 2024; Parikh *et al.*, 2020]
	Formula Generation	[Chen *et al.*, 2021; Joshi *et al.*, 2024; Wang *et al.*, 2025; Zhao *et al.*, 2024]

Table 1: Taxonomy of tasks on LLMs for Spreadsheet Intelligence.

relational representation is necessary for the use of relational integration tools for advanced data analysis.

To test different techniques, the **EUSES** [Fisher and Rothermel, 2005] spreadsheet corpus has been a favored choice. However, it does not closely align with real-world business scenarios. Hermans and Murphy-Hill [2015] address this limitation and propose **Enron**, a much larger collection of over 15,000 spreadsheets retrieved from internal emails. In addition, **DECO** [Koci *et al.*, 2019] is a dataset of annotated spreadsheet files for table recognition, extracted from Enron. The authors not only annotate the layout roles for non-empty cells but also mark the table borders.

3.1 Decomposition from Structured Formats

Table recognition is a rich research field that predates LLM applications to spreadsheet data. Yet, due to the diverse table formats found in spreadsheets, existing methods still lack reliability; presently, no single solution performs robustly across all table types [Fang *et al.*, 2021].

Previously, Chen and Cafarella [2014] introduce a two-phase semiautomatic system that automatically extracts tabular data from spreadsheets, enabling relational spreadsheet integration. The extractor uses cues from the graphical style and metadata to extract the spreadsheet data. Next, an interactive repair module identifies similar regions in different spreadsheets across the dataset, allowing the user to manually correct other possible extraction errors.

Users often organize similar data and computations by repeatedly copying the same block of cells. These blocks use identical or similar formats, representing the fundamental structure of spreadsheets. Without a proper tool to extract these blocks of cells, downstream spreadsheet analysis tools, such as those for data integration and fault detection, cannot robustly perform advanced data analysis. To address this, Dou *et al.* [2018] define a block and its relevant neighboring blocks as an expandable group and propose *ExpCheck* to extract these structures in spreadsheets. Based on continuous blocks with the same format, *ExpCheck* inspects the format of each block and then classifies them into expandable groups.

3.2 Decomposition from Unstructured Formats

Furthermore, users may share spreadsheets in PDF format. Extracting structured information from complex, seemingly unstructured documents remains a fundamental yet challenging problem across modern industrial applications. These PDF documents are often untagged, making it more challenging to recognize tables, charts, and notes. Shigarov *et al.* [2016] propose a system to identify tables in untagged PDF documents. However, it requires adjusting hyperparameters and ad-hoc heuristic knowledge to retrieve these table cells.

The inherent complexity of irregular table layouts, cross-references, and heterogeneous formats also causes conventional Optical Character Recognition (OCR) methods to fail. To address these limitations, *AID-agent* [Li *et al.*, 2025] leverage the synergy of OCR systems and LLMs to enhance text understanding and extraction capabilities. With a customizable toolset for handling complex document structures, it allows for domain- and task-specific customization. Empirical evaluations in real-world scenarios demonstrated that *AID-agent* substantially outperforms traditional OCR- and LLM-based extraction methods.

3.3 Formatting Elements Extraction

Formatting elements, such as background color and borders, encode valuable visual and contextual hints that help users understand and process spreadsheets more effectively. They also define individual regions and encode the semantic meanings of the table. However, due to token constraints, these elements are often neglected in research works [Dong *et al.*, 2024]. *SheetAgent* [Chen *et al.*, 2025] supports limited formatting options for the Editing & Handling task. Presently, *Auto-Formula* [Chen *et al.*, 2024] and *SheetCopilot* [Li *et al.*, 2023] are the only methods that work with formatting infor-

mation. Notably, *SheetCopilot* supports a wide range of options, such as customizing fonts, setting data types, merging cells, and resizing rows/columns.

4 Pre-tokenization

Spreadsheet data shares several aspects with tabular data while introducing its own challenges; an important challenge is that a worksheet can consist of multiple tables in different locations. Moreover, feeding the entire spreadsheet directly as input can quickly overflow the context window of LLMs. Even if the tables are extracted separately, understanding and encoding the relationships between them, or selecting which ones are relevant for a specific task, remains a challenge. This section outlines various techniques for table alignment, which are independent and can be combined in different ways to achieve a satisfactory global spreadsheet encoding solution.

4.1 Modalities

As LLMs are mostly pre-trained on textual inputs, a reasonable first step is to serialize all structured data into text data. Recently, following the development of Large Vision-Language Models (VLMs), several works additionally leverage images that are carefully extracted from spreadsheets.

Text Modality

A straightforward approach is to use input tables directly in formatting languages such as CSV, JSON, HTML, Markdown, Pandas DataFrame, or to format them as a list of dictionaries [Singha *et al.*, 2023]. *MediTab* [Wang *et al.*, 2024a] pairs column names with cell values, while *SheetEncoder* [Dong *et al.*, 2024] pairs cell locations with cell values. Alternatively, Yu *et al.* [2023] describe the tables as natural language sentences using pre-defined templates based on column names and cell values. Crucially, LLM performance is known to be sensitive to the input format. While JSON and DataFrame formats are better suited for table manipulation with programming languages, Fang *et al.* [2024] show that General Pre-trained (GPT) models perform better with HTML and XML code for Question-Answering and Fact-Verification tasks. Additionally, Sui *et al.* [2024] provide evidence that, generally, markup languages outperform other formats for common tabular tasks with GPT models. They hypothesize that this is due to GPT models being exposed more frequently to tables in HTML/XML formats, as a large portion of their training data is obtained from web scraping.

Nonetheless, Chen [2023] show that, despite not being pre-trained on table-specific corpora, LLMs exhibit strong capabilities for complex reasoning over table structures when combined with Chain-of-Thought (CoT) prompting and 1-shot in-context learning (ICL). This suggests that LLMs could play a key role in table reasoning tasks. Unlike generic reasoning, it requires models to understand both queries in free-form text and semi-structured spreadsheet data.

While CoT methods reason on textual representations, Wang *et al.* introduce *Chain-of-Table* to perform reasoning directly on tabular representations. They augment the prompt with ICL and construct a tabular reasoning chain by iteratively updating the source table. At each step, the model plans the next tabular operation based on the current intermediate table. As the evolving sequence of table states encodes the reasoning trajectory for the task, *Chain-of-Table* supports more accurate and reliable predictions.

Image Modality

Another less common but recent direction is to investigate how VLMs handle tabular data when their tables are presented as images rather than in text formats. Deng *et al.* [2024] systematically evaluate both purely textual LLMs and multimodal LLMs across a range of table-related tasks. The authors demonstrate that LLMs generally maintain performance when presenting tables as rendered images. Similarly, Borisova *et al.* [2025] evaluate various LLMs and VLMs on *TableEval* using multiple table representation formats: image, LaTeX, XML, HTML, and dictionary. Empirically, image-based table representations outperform text-based representations across various metrics. This finding is consistent with prior research, which has shown that models often perform on par or better with visual table inputs.

Unfortunately, Xia *et al.* [2024] show that VLMs do not work as well with images extracted directly from spreadsheet data. They assess different VLMs on OCR and visual format recognition tasks, which also include spreadsheet table detection as an aggregate task. They also designed three spreadsheet-to-image variants by adjusting the column widths, modifying the style of the tables, or applying augmentations to probe model behavior more precisely. Experimental results show that, although VLMs exhibit promising OCR capabilities, they frequently suffer from cell omissions and misalignment and perform poorly on format recognition, highlighting the need for future approaches that improve spreadsheet intelligence.

4.2 Context Enrichment

Without pre-processing, compression, or enhancement, using serialized spreadsheets as input quickly skyrockets the token counts, which ultimately harms performance and costs. Their tables contain different columns for various feature types or even a series of values as column names. Rows can be independent records or correlated time series data. To holistically understand such data, it is crucial to fit the encoded tables within the context window. It is easy to serialize a small table and then append it to the prompt. In contrast, for larger tables, short context length and the quadratic complexity of self-attention remain unsolved challenges.

Prompt Optimization

To address these issues, naive methods involve truncating the input given a maximum sequence length. Fu *et al.* [2025] introduced a prompt compression method for an LLM tool-calling on edge devices. Dong *et al.* [2024] suggested that encoding categorical features would significantly help reduce the token count. *ColNet* [Chen *et al.*, 2019] embeds the semantics of tables by predicting column types. *ForTaP* [Cheng *et al.*, 2022a] later adapts BERT architecture and achieves SOTA performance on HiTab [Cheng *et al.*, 2022b].

A more recent common strategy is to search for and retrieve only relevant records, then feed the examples as an ICL approach. Fang *et al.* [2024] reported that changing the

prompt from one-shot to zero-shot decreased the accuracy by 30.38% on all tasks. While increasing from one-shot to two-shot support could improve the model, further increases did not enhance its performance [Chen, 2023].

Retrieval-Augmented Generation (RAG)

Although LLMs are general models, by incorporating relevant tables, columns, rows, or cells into the prompts, users can guide the model to generate more specific answers. A vital task of RAG is extracting the most relevant information from an extensive database to better guide the LLMs. Since RAG only works well with a carefully structured database, adapting this approach requires accurate table extraction.

cTBLS [Sundar and Heck, 2023] is a dense table retrieval model that uses a dual-encoder to rank table cells according to their relevance. Based on their ROUGE scores, the knowledge sources are then used to augment the prompt. *RITT* [Zhou *et al.*, 2025] filters relevant records from the table by executing Python code generated by the LLM.

Transforming a table to a graph or tree is a feasible but less commonly explored method. However, the major downside is that these structures still have to be converted back to text when working with LLMs, which might lose their structural and relational information. Zhao *et al.* [2023] converted the tables into trees and used a tuple to save the position, value, and hierarchical structure of the cell. *LERGV* [Shi *et al.*, 2021] is a table-based fact verification framework assisted by evidence retrieval. A graph encodes the relationship between a source entity and the retrieved evidence. Then, a verification module analyzes this graph to classify the final relation.

4.3 Advanced Transformations

Spreadsheet-to-SQL

Despite the limited number of works on mapping spreadsheets to relational databases, using SQL as a medium in tabular tasks is an emerging research direction, as tabular data is closely tied to the structure of relational databases. Compared to RAG methods, SQL-based retrieval offers more accurate and lossless data extraction from tables, which ultimately leads to hallucination-free output.

Table-to-SQL is a preprocessing or data cleaning task. Here, the system attempts to generate an SQL-friendly format directly from the table itself. The tools need to understand the table structure, its column types, and perform table operations such as pivoting from a wide to a long table, removing duplicated or empty rows and columns, etc. In standardized relational tables, rows represent entities and columns represent attributes, forming a standard practice in database systems. However, many tables in real life deviate from this norm. More than 30% of tables are not relational and require substantial restructuring before they can be effectively interacted with SQL tools [Li *et al.*, 2024]. According to a large number of support requests in community forums, this multi-step transformation is unfortunately non-trivial and challenging for both non-technical and professional users. *Auto-Tables* [Li *et al.*, 2024] alleviates this burden by streamlining the process through multi-step transformations using Python or R to standardize tables into a relational form, enabling seamless downstream analytics and eliminating the need for users to handcraft complex transformation code.

As noted above, spreadsheets do not contain solely tabular data but are often accompanied by important text data, such as notes or comments. Text-to-SQL refers to intent understanding, in which the LLM converts natural language prompts into SQL queries. This requires understanding intent and mapping words and phrases to the correct tables or columns in order to make proper use of SQL functions such as JOIN, GROUP BY, etc. *StructGPT* [Jiang *et al.*, 2023] further utilizes tables, relational databases, and knowledge graphs to generalize the representation across different modalities.

Agentic Approaches

LLMs struggle with direct data manipulation in spreadsheets, especially when performing complex mathematical operations or fetching particular numerical values. To overcome their lack of robustness in handling scattered numerical data, agentic LLMs emerged as a solution by extending the capabilities of LLMs with a predefined and orchestrated toolset. Existing agents employ various techniques to decompose complex tasks, leverage external tools, and utilize sampling or iterative refinement strategies to achieve their objectives. Inspired by reasoning paradigms such as CoT, LLMs can be guided to break down a complex problem into smaller, more manageable subtasks. For larger tables, Ye *et al.* [2023] demonstrated the effectiveness of dividing a large dataset into compact and interpretable segments and then solving these sub-problems individually. However, agents do not always succeed on the first attempt. To improve orchestration robustness, *ReAct* [Yao *et al.*, 2023] is a widely-used method. In this iterative framework, the agent repeatedly analyzes the requirements, generates action plans, executes them, reflects on the results, and refines its plans until the final result is achieved. Extending beyond traditional table reasoning, *TAP4LLM* [Sui *et al.*, 2024] further adapts agents to handle sampling and augmenting tables, highlighting the versatility of agents in different tasks. The design of the available tools critically determines the effectiveness of an agent. Beyond basic primitive operations, various systems extend the toolset to include domain-specific functions. For instance, *SheetCopilot* [Li *et al.*, 2023] integrates spreadsheet APIs as callable actions; *SheetAgent* [Chen *et al.*, 2025], a self-evolving multi-agent system, aims to automatically solve a wide range of spreadsheet reasoning and manipulation tasks.

5 Downstream Tasks & Benchmarks

Spreadsheet intelligence encompasses a wide range of tasks for real-world applications. LLM-based approaches represent the prevailing research direction for automating complex data analysis workflows and enabling natural language interaction with enterprise data systems. In this section, we list different tasks and their corresponding benchmarks.

5.1 Question Answering (QA)

QA is among the most common tasks related to spreadsheet understanding. Here, models answer user queries about spreadsheet data by mapping user questions to relevant spreadsheet cells. This involves a wide range of actions, for

example, search, comparison, aggregation, and arithmetic operations. Unlike tabular QA, spreadsheet QA involves reasoning across multiple sheets with nested tables, understanding cell dependencies, and formulas. It reflects practical business workflows, where users consult spreadsheets for insights such as summaries, comparisons, or pattern analysis.

Recent research focuses on equipping LLMs with external tools and domain-specific actions to improve their effectiveness in spreadsheet environments. This approach transforms LLMs into interactive agents capable of understanding and reasoning with spreadsheets to synthesize the final answer. To further enhance multi-step reasoning across complex sheets, prompting patterns such as CoT guide the LLM to decompose the query into manageable sub-tasks before deriving the final output. In some scenarios, QA also involves semantic understanding of cell/region references and aggregation logic.

DSBench [Jing *et al.*, 2025] is a comprehensive benchmark for evaluating LLM systems on realistic spreadsheet reasoning tasks. It comprises 466 data analysis and 74 data modeling challenges, crawled from Eloquence and Kaggle competitions. The data analysis challenges require agents to deal with long contexts and multimodal task descriptions and reason over large Excel files with multi-table schemas.

SpreadsheetQA [Dong *et al.*, 2024] focuses on referring to the correct cell location instead of the final answer. This requires models to take cell indices in consideration, challenging code-based solutions that work solely with DataFrames.

Since spreadsheet-specific benchmarks remain limited, researchers can still benefit from sophisticated Tabular QA benchmarks. **HiTab** [Cheng *et al.*, 2022b] extends beyond flat tables and takes into account hierarchical tables, involving complex multi-level indexing and implicit relationships. Empirical results indicate that HiTab is highly challenging for existing methods and thus can serve as a helpful benchmark for advancing research in table reasoning.

WikiMixQA [Foroutan *et al.*, 2025] consists of 1000 multiple-choice questions that evaluate multi-modal reasoning. In addition to previous benchmarks, it focuses on assessing complex and advanced reasoning that demands synthesizing information across multiple tables and charts. The authors highlight the persistent limitations of current LLMs in long-context, multi-modal reasoning capabilities.

Spider [Yu *et al.*, 2018] is a comprehensive text-to-SQL dataset of more than 10K questions and roughly 6K complex SQL queries. Thanks to the challenging text-to-SQL task across domains, where SQL patterns and database schemas differ between training and test sets, it forces models to generalize to unseen samples rather than memorize templates.

5.2 Fact Verification (FV)

FV aims to determine the authenticity of a given claim by assessing it based on reliable evidence sources. In domains such as finance or law, where domain-grounded retrieval and precise numerical reasoning are critical, FV is necessary to ensure that the output is trustworthy. Traditionally, this task focuses on misinformation detection and knowledge-grounded reasoning. It requires systems to retrieve supporting evidence and classify claims as true, false, or unsupported. Early FV systems relied on information retrieval to perform supervised classification. LLM reimagined the task by augmenting context with evidence synthesis through an end-to-end reasoning chain using natural language prompts.

Despite these advances, the possibility of hallucination remains a key challenge. LLMs struggle with fine-grained evidence attribution and are sensitive to the phrasing of prompts. Modern LLM-based FV systems perform RAG, where models both retrieve evidence and articulate natural language justifications. This improves truthfulness and provides verifiable, interpretable reasoning chains. Hence, LLM-based FV serves as a testbed for developing trustworthy AI systems.

FEVEROUS [Aly *et al.*, 2021] evaluates whether a given claim is supported by evidence from Wikipedia. Compared to *FEVER* [Thorne *et al.*, 2018], it introduces longer, more complex claims and tables as additional sources of evidence.

5.3 Editing and Handling

File editing and handling with LLMs typically relies on a tool-augmented architecture where the model translates textual commands into file operations. As LLMs cannot directly access the file system, it is crucial to equip them with specialized tools that accept structured parameters describing the target files, the desired changes, and then execute and return feedback for further steps. Recent works highlight several common patterns in file editing workflows, including incremental context building followed by precise modification and verification steps. Here, LLMs execute multi-step edits, chart generation, formatting, and so on using natural language commands or through agent frameworks.

SheetCopilot [Li *et al.*, 2023] is a benchmark comprising 221 spreadsheet editing tasks and an automated evaluation pipeline to assess software control capabilities. The integration of spreadsheet APIs and a Python executor allows LLMs to retrieve cell values, compute groups, and generate formulas in a structured manner. The authors defined a set of atomic actions that perform core spreadsheet functionalities and can be composed to execute complex operations.

SheetRM [Chen *et al.*, 2025] focuses on long-horizon, multi-category spreadsheet tasks. It is suitable for evaluating agentic systems on file manipulation with reasoning under realistic, real-world challenges.

SODBench [Indika and Molybog, 2025] focuses on generating explanations from spreadsheet manipulation code, thereby bridging the gap between the system's low-level operations and the user's high-level intents. The authors constructed a benchmark of 111 spreadsheet manipulation codes, aligning with natural language summaries.

5.4 Data Analysis

Tabular data analysis is pivotal across many domains. LLMs have recently demonstrated strong potential for assisting with these tasks; however, existing research predominantly targets QA or FV problems, overlooking advanced analyses such as forecasting and chart generation.

Text2Analysis [He *et al.*, 2024] is a benchmark that focuses on complex analytical tasks, which expand beyond SQL operations and require deep reasoning over tables. It contains 2,249 query–result pairs over 347 tables, covering

a broad spectrum of advanced analysis scenarios. Evaluations demonstrate that Text2Analysis poses significant challenges for current systems, highlighting a necessity for future research in LLM-based tabular data analysis.

ToTTo [Parikh *et al.*, 2020] is a table-to-text dataset consisting of more than 120,000 samples. The task is to produce a descriptive sentence given a table crawled from Wikipedia and a set of table cells. The empirical study reveals that existing methods often generate phrases that contradict the table. This suggests that ToTTo can be a reliable benchmark for high-quality conditional text generation.

5.5 Formula Generation

Writing formulas in applications, predominantly Microsoft Excel and Google Sheets, is a standard part of data analysis workflows; yet, composing correct formulas is often tedious and error-prone, especially for complex operations. Formula generation with LLMs focuses on translating intents into executable spreadsheet formulas, thereby relieving users of the need to memorize multiple function names, argument orders, and nesting patterns. Typical intents involve conditional aggregation, data lookup, and multi-criteria filtering. These actions require LLMs to understand not only the user's description but also the underlying sheet layout and data types.

In practice, commercial tools such as Microsoft Excel and Google Sheets embed their Copilot, Gemini LLMs directly into the spreadsheet interface. Users can prompt the desired action, and then the LLMs suggest formulas or automatically generate formula columns and rows. These systems often pair formula suggestions with natural language explanations, encouraging users to verify and refine the generated expressions before applying them at scale.

NL2Formula [Zhao *et al.*, 2024] aims to generate executable formulas in a given spreadsheet from a natural language query. It is a large-scale dataset containing more than 70,000 NL–formula pairs associated with over 21,000 tables and covering 37 distinct formula function types.

FormulaQA [Wang *et al.*, 2025] is a formula-annotated dataset for existing tabular QA benchmarks. The authors construct a "Direct" variation that requires LLMs to gain proficiency in tabular QA and a smaller "CoT" variation that focuses on the reasoning-then-generation pattern.

Chen *et al.* [2021] constructed more than 800K samples from 46K Google Sheets files. Although the formula windows are relatively small (at most 10 cells away), the authors still found it challenging to generate entirely correct formulas. Additionally, various frameworks have extracted formulas from large corpora, such as EUSES [Fisher and Rothermel, 2005] and Enron [Hermans and Murphy-Hill, 2015].

6 Conclusion & Future Directions

Recent works have shown notable progress in addressing the inherent challenges of representing and reasoning about spreadsheets. Commercial and research solutions can now achieve promising results in realistic tasks. Through the integration of specialized statistical tools to LLM systems, developers can enhance numerical accuracy and mitigate hallucinations in complex financial contexts. Nevertheless, the path toward a reliable, autonomous spreadsheet intelligence system requires further advancements and considerations, as issues regarding accuracy, explainability, and robust grounding remain unsolved. Regarding the monitoring of complex reasoning steps, benchmarks reflecting real-world scenarios with strict validation can help mitigate persistent hallucinations, especially numerical ones.

First, there are still spreadsheet **tasks, where LLMs lag behind human performance**. For instance, in DSBench [Jing *et al.*, 2025], GPT-4o achieves 28.11% accuracy on the QA task, compared to human performance of 64.04%. The authors also adapt an agentic workflow, but it only achieves 34.12% accuracy. From our point of view, refining the decomposition stage and improving multimodal understanding can help models achieve better performance. Moreover, practical spreadsheet use in corporate environments involves interconnected workbooks; accommodating multi-sheet and multi-file functionality is another dimension for expansion.

The above visions require the curation of **comprehensive benchmarks for end-to-end systems**. Currently, there is no gold-standard benchmark for spreadsheet intelligence. As a consequence, methods are evaluated on different ad-hoc datasets, making in-depth comparisons difficult or even impossible. Future benchmarks should be available to test systems with messy, real-world-like spreadsheets, comprehensively from the decomposition stage to downstream tasks.

One of the unexplored directions is to **leverage generic tabular reasoning models, including Tabular Foundation Models** (TFMs). For example, *TabPFN* [Hollmann *et al.*, 2025] is among the most potent TFMs. Being trained on several million synthetic tabular datasets, it can encode new tabular data without fine-tuning, and it has been shown to outperform classical machine learning methods by a significant margin. However, its capability is limited to classification and regression tasks. Combining the strengths of TFMs for tabular data encoding with the reasoning abilities of LLMs is, thus, a promising direction for future work.

Finally, to ensure understanding and trust in LLM-generated answers, it is necessary to address **explainability specifically for spreadsheets**, which requires methods capable of producing explanations across modalities (text, images, tabular, etc.). In high-risk domains such as finance, achieving reliable references to the correct cells is crucial. Along with interpretability, monitoring explainability allows for human intervention when the model is not confident about the output. Unfortunately, in the majority of cases, the best performing frameworks use closed-source LLMs (e.g., GPT-4o), making it impossible to examine the underlying reasoning process.

In summary, through this survey, we aim to bring structure to the research field of spreadsheet intelligence with LLMs, forming a foundation for reproducible and scalable spreadsheet understanding systems. We propose a unified problem formulation as a consistent and generic flow in 3 stages. Based on this workflow, we organize existing research works into a taxonomy of tasks, covering both pre-processing tasks and downstream tasks. By surveying an extensive range of models, methods, and benchmarks, we study and discuss the current landscape of spreadsheet intelligence, highlight open challenges, and point out relevant future research directions

to enhance the ability of LLMs in spreadsheet tasks.

Acknowledgments

This research is supported, in whole or in part, by a partnership project of the University of Luxembourg.

References

[Aly *et al.*, 2021] Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the 4th FEVER Workshop*, 2021.

[Birch *et al.*, 2017] David Birch, David Lyford-Smith, and Yike Guo. The future of spreadsheets in the big data era. In *Proceedings of the EuSpRIG 2017 Conference*, 2017.

[Borisova *et al.*, 2025] Ekaterina Borisova, Fabio Barth, Nils Feldhus, Raia Abu Ahmad, Malte Ostendorff, Pedro Ortiz Suarez, Georg Rehm, and Sebastian Möller. Table understanding and multimodal LLMs: A cross-domain case study on scientific vs. non-scientific data. In *Proceedings of the 4th Table Representation Learning Workshop*, 2025.

[Chen and Cafarella, 2014] Zhe Chen and Michael Cafarella. Integrating spreadsheet data via accurate and low-effort extraction. In *Proceedings of ACM SIGKDD*, 2014.

[Chen *et al.*, 2019] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles Sutton. Colnet: Embedding the semantics of web tables for column type prediction. *Proceedings of AAAI*, Jul. 2019.

[Chen *et al.*, 2021] Xinyun Chen, Petros Maniatis, Rishabh Singh, Charles Sutton, Hanjun Dai, Max Lin, and Denny Zhou. Spreadsheetcoder: Formula prediction from semi-structured context. In *Proceedings of the 38th ICML*, 2021.

[Chen *et al.*, 2024] Sibei Chen, Yeye He, Weiwei Cui, Ju Fan, Song Ge, Haidong Zhang, Dongmei Zhang, and Surajit Chaudhuri. Auto-formula: Recommend formulas in spreadsheets using contrastive learning for table representations. *Proc. ACM Manag. Data*, May 2024.

[Chen *et al.*, 2025] Yibin Chen, Yifu Yuan, Zeyu Zhang, Yan Zheng, Jinyi Liu, Fei Ni, Jianye Hao, Hangyu Mao, and Fuzheng Zhang. Sheetagent: Towards a generalist agent for spreadsheet reasoning and manipulation via large language models. In *Proceedings of the ACM WWW*, 2025.

[Chen, 2023] Wenhu Chen. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, May 2023.

[Cheng *et al.*, 2022a] Zhoujun Cheng, Haoyu Dong, Ran Jia, Pengfei Wu, Shi Han, Fan Cheng, and Dongmei Zhang. FORTAP: Using formulas for numerical-reasoning-aware table pretraining. In *Proceedings of the 60th ACL*, 2022.

[Cheng *et al.*, 2022b] 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 *Proceedings of the 60th ACL*, May 2022.

[Deng *et al.*, 2024] Naihao Deng, Zhenjie Sun, Ruiqi He, Aman Sikka, Yulong Chen, Lin Ma, Yue Zhang, and Rada Mihalcea. Tables as texts or images: Evaluating the table reasoning ability of LLMs and MLLMs. In *Findings of ACL 2024*, August 2024.

[Dong *et al.*, 2024] Haoyu Dong, Jianbo Zhao, Yuzhang Tian, Junyu Xiong, Mengyu Zhou, Yun Lin, José Cambronero, Yeye He, Shi Han, and Dongmei Zhang. Encoding spreadsheets for large language models. In *Proceedings of EMNLP 2024*, November 2024.

[Dou *et al.*, 2018] Wensheng Dou, Shi Han, Liang Xu, Dongmei Zhang, and Jun Wei. Expandable group identification in spreadsheets. In *Proceedings of the 33rd ACM/IEEE ASE*, September 2018.

[Fang *et al.*, 2021] Jing Fang, Prasenjit Mitra, Zhi Tang, and C. Lee Giles. Table header detection and classification. *Proceedings of AAAI*, Sep. 2021.

[Fang *et al.*, 2024] Xi Fang, Weijie Xu, Fiona Anting Tan, Ziqing Hu, Jiani Zhang, Yanjun Qi, Srinivasan H. Sengamedu, and Christos Faloutsos. LLMs on tabular data: Prediction, generation, and understanding - a survey. *Transactions on Machine Learning Research*, 2024.

[Fisher and Rothermel, 2005] Marc Fisher and Gregg Rothermel. The euses spreadsheet corpus: a shared resource for supporting experimentation with spreadsheet dependability mechanisms. In *Proceedings of the First Workshop on End-User Software Engineering*, 2005.

[Foroutan *et al.*, 2025] Negar Foroutan, Angelika Romanou, Matin Ansaripour, Julian Martin Eisenschlos, Karl Aberer, and Rémi Lebret. WikiMixQA: A multimodal benchmark for question answering over tables and charts. In *Findings of ACL 2025*, July 2025.

[Fu *et al.*, 2025] Yicheng Fu, Raviteja Anantha, and Jianpeng Cheng. CAMPHOR: Collaborative agents for multi-input planning and high-order reasoning on device. In *Proceedings of the 1st REALM Workshop*, July 2025.

[He *et al.*, 2024] Xinyi He, Mengyu Zhou, Xinrun Xu, Xiaojun Ma, Rui Ding, Lun Du, Yan Gao, Ran Jia, Xu Chen, Shi Han, Zejian Yuan, and Dongmei Zhang. Text2analysis: a benchmark of table question answering with advanced data analysis and unclear queries. In *Proceedings of AAAI'24/IAAI'24/EAAI'24*, 2024.

[Hermans and Murphy-Hill, 2015] Felienne Hermans and Emerson Murphy-Hill. Enron's spreadsheets and related emails: A dataset and analysis. 2015.

[Hermans *et al.*, 2012] Felienne Hermans, Martin Pinzger, and Arie van Deursen. Detecting code smells in spreadsheet formulas. In *2012 28th IEEE ICSM*, 2012.

[Hollmann *et al.*, 2025] Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 2025.

[Indika and Molybog, 2025] Amila Indika and Igor Molybog. Sodbench: A large language model approach to documenting spreadsheet operations, 2025.

[Jaimovitch-López *et al.*, 2022] Gonzalo Jaimovitch-López, Cèsar Ferri, José Hernández-Orallo, Fernando Martínez-Plumed, and María José Ramírez-Quintana. Can language models automate data wrangling? *Mach. Learn.*, 2022.

[Jiang *et al.*, 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 *Proceedings of EMNLP 2023*, 2023.

[Jing *et al.*, 2025] Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. DSBench: How far are data science agents from becoming data science experts? In *The Thirteenth ICLR*, 2025.

[Joshi *et al.*, 2024] Harshit Joshi, Abishai Ebenezer, José Cambronero Sanchez, Sumit Gulwani, Aditya Kanade, Vu Le, Ivan Radiček, and Gust Verbruggen. Flame: a small language model for spreadsheet formulas. In *Proceedings of AAAI'24/IAAI'24/EAAI'24*, 2024.

[Koci *et al.*, 2019] Elvis Koci, Maik Thiele, Josephine Rehak, Oscar Romero, and Wolfgang Lehner. Deco: A dataset of annotated spreadsheets for layout and table recognition. In *2019 ICDAR*, 2019.

[Li *et al.*, 2023] Hongxin Li, Jingran Su, Yuntao Chen, Qing Li, and Zhaoxiang Zhang. Sheetcopilot: bringing software productivity to the next level through large language models. In *Proceedings of the 37th NeurIPS*, 2023.

[Li *et al.*, 2024] Peng Li, Yeye He, Cong Yan, Yue Wang, and Surajit Chaudhuri. Auto-tables: Relationalize tables without using examples. *SIGMOD Rec.*, May 2024.

[Li *et al.*, 2025] Bin Li, Jannis Conen, and Felix Aller. AID-agent: An LLM-agent for advanced extraction and integration of documents. In *Proceedings of the 1st Workshop for Research on Agent Language Models*, July 2025.

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

[Shi *et al.*, 2021] Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. Logic-level evidence retrieval and graph-based verification network for table-based fact verification. In *Proceedings of EMNLP 2021*, November 2021.

[Shigarov *et al.*, 2016] Alexey Shigarov, Andrey Mikhailov, and Andrey Altaev. Configurable table structure recognition in untagged pdf documents. In *Proceedings of the 2016 ACM Symposium on Document Engineering*, 2016.

[Singha *et al.*, 2023] Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. Tabular representation, noisy operators, and impacts on table structure understanding tasks in LLMs. In *NeurIPS 2023 Second Table Representation Learning Workshop*, 2023.

[Sui *et al.*, 2024] 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. In *Findings of EMNLP 2024*, November 2024.

[Sundar and Heck, 2023] Anirudh S. Sundar and Larry Heck. cTBLS: Augmenting large language models with conversational tables. In *Proceedings of the 5th NLP4ConvAI Workshop*, 2023.

[Thorne *et al.*, 2018] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of NAACL 2018*, June 2018.

[Wang *et al.*, 2024a] Zifeng Wang, Chufan Gao, Cao Xiao, and Jimeng Sun. Meditab: scaling medical tabular data predictors via data consolidation, enrichment, and refinement. In *Proceedings of the Thirty-Third IJCAI*, 2024.

[Wang *et al.*, 2024b] Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. Chain-of-table: Evolving tables in the reasoning chain for table understanding. In *The Twelfth ICLR*, 2024.

[Wang *et al.*, 2025] Zhongyuan Wang, Richong Zhang, Zhijie Nie, and Hangyu Mao. General table question answering via answer-formula joint generation, 2025.

[Xia *et al.*, 2024] Shiyu Xia, Junyu Xiong, Haoyu Dong, Jianbo Zhao, Yuzhang Tian, Mengyu Zhou, Yeye He, Shi Han, and Dongmei Zhang. Vision language models for spreadsheet understanding: Challenges and opportunities. In *Proceedings of the 3rd ALVR Workshop*, August 2024.

[Yao *et al.*, 2023] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

[Ye *et al.*, 2023] 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 *Proceedings of the 46th International ACM SIGIR*, 2023.

[Yu *et al.*, 2018] 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 *Proceedings of EMNLP*, 2018.

[Yu *et al.*, 2023] Bowen Yu, Cheng Fu, Haiyang Yu, Fei Huang, and Yongbin Li. Unified language representation for question answering over text, tables, and images. In *Findings of ACL 2023*, July 2023.

[Zhao *et al.*, 2023] 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 *Proceedings of EMNLP 2023*, December 2023.

[Zhao *et al.*, 2024] Wei Zhao, Zhitao Hou, Siyuan Wu, Yan Gao, Haoyu Dong, Yao Wan, Hongyu Zhang, Yulei Sui,

and Haidong Zhang. NL2Formula: Generating spreadsheet formulas from natural language queries. In *Findings of EACL 2024*, March 2024.

[Zhou *et al.*, 2025] Wei Zhou, Mohsen Mesgar, Heike Adel, and Annemarie Friedrich. RITT: A retrieval-assisted framework with image and text table representations for table question answering. In *Proceedings of the 4th Table Representation Learning Workshop*, July 2025.