Making Multimodal LLMs Reliable Chart Data Extractors: A Benchmark and Training Framework

Yuchen He*
Zhejiang University
Hangzhou, Zhejiang, China
heyuchen@zju.edu.cn

Peizhi Ying
Zhejiang University
Hangzhou, Zhejiang, China
yingpeizhi@zju.edu.cn

Liqi Cheng
Zhejiang University
Hangzhou, Zhejiang, China
lycheecheng@zju.edu.cn

Kuilin Peng
Guangdong University of Technology
Guangzhou, Guangdong, China
kuilinpeng3@gmail.com

Yuan Tian
Zhejiang University
Hangzhou, Zhejiang, China
yuantian@zju.edu.cn

Dazhen Deng†
Zhejiang University
Hangzhou, Zhejiang, China
dengdazhen@zju.edu.cn

Yingcai Wu
Zhejiang University
Hangzhou, Zhejiang, China
ycwu@zju.edu.cn

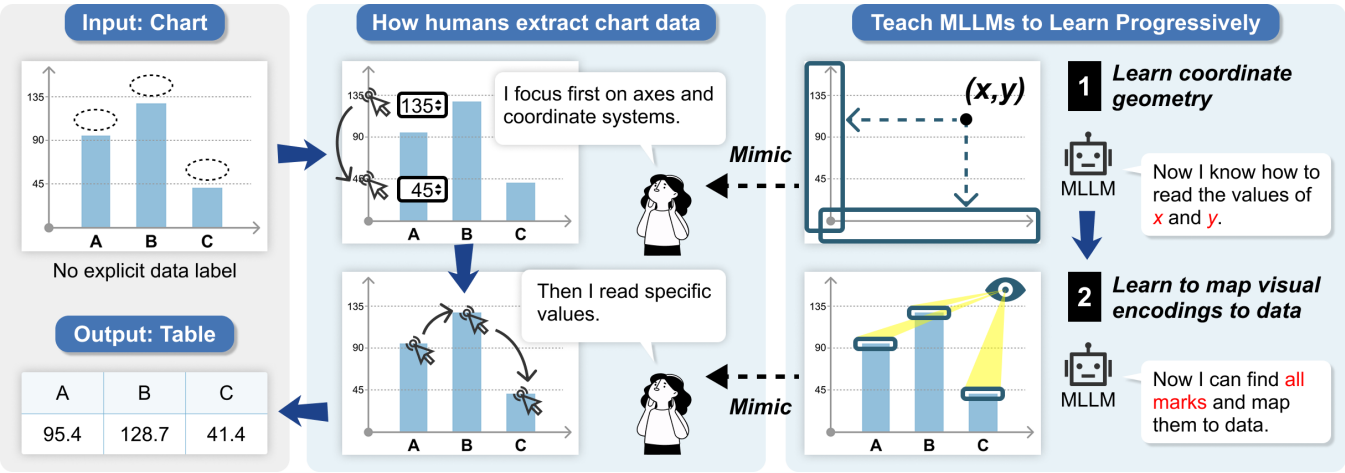

Figure 1: Chart data extraction aims to recover the underlying data table from a chart image. Although existing interactive tools and mixed-initiative workflows can produce reliable results, they fall short in terms of efficiency or generalizability. Current MLLMs are a potential solution, but they struggle to infer precise values when explicit data labels are absent. We introduce a two-stage training framework that guides MLLMs to learn in a progressive, human-like manner: first, they learn coordinate geometry; then, they map visual encodings to data.

Abstract

Chart data extraction, which reverse-engineers data tables from chart images, is essential for reproducibility, analysis, retrieval, and redesign. Existing interactive tools are reliable but tedious, and mixed-initiative systems, while more efficient, lack generalizability. Recent multimodal large language models (MLLMs) offer a unified interface for chart interpretation, yet their ability to extract accurate data tables, especially without visible labels, remains unclear. We build a benchmark featuring diverse real-world charts without data labels to evaluate this capability. Results show that, while current MLLMs reliably reconstruct table structures, they struggle with precise value recovery. To address this, we revisit chart data extraction from a human-centered perspective and argue that extraction should follow a progressive learning process similar to how people read charts. Our training framework substantially improves numerical accuracy, achieving state-of-the-art performance with a

*All authors are affiliated with or completed this work during their internship at the State Key Lab of CAD&CG, Zhejiang University.
†Dazhen Deng is the corresponding author.

CHI '26, Barcelona, Spain

ACM ISBN 979-8-4007-2278-3/26/04
https://doi.org/10.1145/3772318.3790721

7B-parameter model. A user study further shows that our model effectively supports mixed-initiative workflows for reliable chart data extraction.

CCS Concepts

• **Human-centered computing → Visualization**.

Keywords

Chart Data Extraction, Multimodal Large Language Models

ACM Reference Format:

Yuchen He, Peizhi Ying, Liqi Cheng, Kuilin Peng, Yuan Tian, Dazhen Deng, and Yingcai Wu. 2026. Making Multimodal LLMs Reliable Chart Data Extractors: A Benchmark and Training Framework. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3772318.3790721

1 Introduction

Charts serve as a primary medium for communicating quantitative information across scientific publications, business reports, and policy documents [24, 28]. Their central role in data communication has made chart understanding and generation a vibrant research area in visualization and human-computer interaction [8, 16, 31, 48, 60, 63, 64, 74]. Among these challenges, chart data extraction, the process of recovering structured data from chart images, remains particularly valuable yet difficult. When charts are published without accompanying data tables, researchers cannot reproduce analyses, verify claims, or reuse the data for meta-analyses, redesign, or accessibility purposes [6, 43, 44].

Based on efficiency and reliability, i.e., how accurately the extracted data reflects the underlying chart data [30], existing approaches for chart data extraction fall into three categories. Fully automatic systems [13, 15, 44] such as ChartOCR [44] and ChartDecoder [13] aim to extract values without user intervention, but often struggle with real-world charts due to reliance on multi-stage parsing pipelines, hand-crafted rules, or brittle visual heuristics. These systems lack an efficient mechanism for user correction. So far, interactive tools [43, 57, 66], such as WebPlotDigitizer [43], remain practitioners' preferred choice because they allow users to explicitly calibrate axes and select visual marks. They are reliable but tedious and time-consuming, and do not scale well to large chart collections.

To reduce manual effort while preserving user control, mixed-initiative systems [20, 31, 50, 55] combine machine learning models and interactive verification. For example, ChartSense [31] integrates chart classification and mark detection into an interface that supports step-by-step refinement. While effective in practice, these systems fundamentally rely on task-specific vision modules, i.e., detectors trained for particular mark types, axis layouts, or chart styles. Such specialized components impose strong assumptions on the input and limit generalizability to broader chart styles [31, 50]. Moreover, since each module handles only one subtask, the extraction workflow becomes fragmented, and users must frequently intervene at multiple stages, reducing scalability when working with large chart collections.

In contrast, recent advances in multimodal large language models (MLLMs) [68, 75, 77] offer a unified interface for chart interaction: a single model can answer questions, describe structures, and generate structured outputs across diverse chart types [19, 24, 48, 67, 71]. This general-purpose capability motivates us to revisit chart data extraction from an HCI perspective: if MLLMs are becoming the interaction layer for many tasks, how should they be guided to better support reliable and scalable chart data extraction workflows?

Recent HCI work on chart interpretation [61] and generation [22, 23] offers an important insight here. These systems show that effective chart understanding, whether for reading or constructing visualizations, often follows a progressive cognitive process rather than a single-step mapping. People first attend to axes and coordinate systems, then recognize visual encodings, and only afterward infer specific values. This perspective motivates us to teach MLLMs to learn chart data extraction progressively, rather than treating it as a single-step perception task. If an MLLM could internalize a similar progression, from understanding visual encodings to inferring precise numerical values, it may overcome the brittleness observed in current end-to-end approaches. Thus, rather than directly training models to map charts to tables or code, we explore whether teaching models to understand visual encodings the way humans do, from simple to complex, can fundamentally enhance accuracy and interpretability in chart data extraction.

Guided by this human-inspired view, we develop **ExChart**, a two-stage training framework that separates foundational visual reasoning from full data extraction. The first stage focuses on coordinate system perception. It trains the model to interpret atomic marks within specific coordinate systems, thereby reinforcing an internal geometric understanding. The second stage builds on this foundation, teaching the model to translate visual encodings into complete data tables without relying on visible labels. This progressive approach allows MLLMs to handle unlabeled charts, a critical real-world scenario, with substantially improved stability and accuracy.

To understand where existing MLLMs fall short and to measure progress enabled by ExChart, we construct **ExChart-Bench**, a benchmark spanning real-world and synthetic charts across diverse styles and encodings. A systematic evaluation of widely used MLLMs on ExChart-Bench reveals a consistent bottleneck: Most models can reproduce table structure reliably but fail to recover accurate values when data labels are absent, confirming findings from prior HCI and chart question answering studies and our practitioner interviews. ExChart-Bench helps surface these gaps and provides a principled way to evaluate both general-purpose MLLMs and specialized models trained with our proposed framework. Across the benchmark, ExChart significantly improves Qwen2.5-VL, achieving the highest accuracy with a model of only 7B parameters. Combined with an interactive validation and correction interface, our user study further shows that MLLM-based extraction can effectively support mixed-initiative workflows, reducing manual workload without compromising reliability. These findings suggest a promising design direction for future HCI systems: grounding MLLMs in human-like visual reasoning and integrating them into interactive workflows may achieve both efficiency and trustworthy numerical extraction at scale. Our project page is available at https://ExChart.github.io/.

Our main contributions are as follows:

- We introduce **ExChart-Bench**, a benchmark for chart data extraction that contains both real-world and synthetic charts to evaluate advanced MLLMs.
- We propose **ExChart**, a human-inspired training framework that enhances coordinate system perception and chart-to-table extraction, significantly enhancing MLLM performance.
- We conduct a user study demonstrating how MLLM-based extraction supports mixed-initiative workflows, showing promise for MLLM-based chart data extraction systems.

2 Related Work

In this section, we review related work on chart data extraction systems and MLLM for chart understanding.

2.1 Chart Data Extraction Systems

Existing chart data extraction systems can be broadly categorized into two groups: fully automatic and interactive, which include mixed-initiative systems. Fully automatic systems such as ChartDecoder [13], ChartOCR [44], and other computer-vision-based pipelines [15] attempt to extract data values directly from chart images by combining OCR, object detection, and structural parsing. However, their dependence on multi-stage pipelines and task-specific components makes them vulnerable to failures when encountering diverse chart styles, distorted scans, aesthetic variations, or charts without visible data labels. The lack of human validation and correction further limits their applicability.

Due to these limitations, interactive tools remain the preferred choice in practical use. Interactive systems such as WebPlotDigitizer [43], PlotDigitizer [57], DataThief [66], and iVoLer [51] remain widely used because they support precise, user-verified extraction. Through manual axis calibration and selection of visual marks, users maintain control over correctness. Yet, this level of manual effort is tedious and does not scale well to large chart collections.

To reduce user effort while preserving reliability, several mixed-initiative systems [20, 55] introduce automation into interactive workflows. For example, ChartSense [31] incorporates deep learning modules for chart classification and mark detection, allowing users to correct errors rather than annotating everything from scratch. These systems exemplify a pragmatic middle ground: automation handles low-level perception tasks, while users validate and refine results. However, their automation components are still limited by task-specific detectors or heuristics. This restricts their ability to generalize to diverse real-world charts. ChartDetective [50] further supports direct manipulation of chart elements and semi-automatic extraction from charts, but requires vector graphics as input. In this work, we explore whether MLLMs can serve as a more flexible and general perception engine for data extraction. This could enable the development of efficient and reliable tools for extracting real-world chart data.

2.2 MLLM for Chart Understanding

With the rise of multimodal large language models (MLLMs), chart interpretation has become more flexible and conversational. A broad range of chart understanding tasks has emerged [24], including chart question answering (CQA) [17, 29, 32, 33, 47–49, 53, 71, 76, 78], captioning or summarization [29, 35, 56, 58, 62, 80], chart-to-code generation [65, 73, 79], and fact-checking [1, 25]. These tasks require models to understand axes, marks, and other encodings, and to integrate visual understanding with natural language instructions.

A growing subset of work investigates generating structured outputs from chart images, such as chart specifications [7, 70, 72, 73, 79]. ChartMimic [73] and ChartCoder [79] study translating chart images into executable specification such as Matplotlib [4] programs, while other approaches [7, 14, 70, 72] reconstruct chart structures or semantic representations. These models focus on the structural interpretation of charts, such as layout and style. However, they ignore the more fundamental ability to accurately interpret the underlying data.

The chart-to-table [29, 47, 72] setting is most closely related to chart data extraction, but prior works typically assume the presence of visible data labels, where the values can be read directly from text when producing tables. In contrast, real-world data extraction rarely includes data labels, requiring models to infer numeric values solely from geometric alignment, an ability that current MLLMs struggle to master [10, 29, 54, 69, 69]. To the best of our knowledge, no existing chart-to-table benchmarks offer diverse, real-world charts while targeting this characteristic. Without such benchmarks, we cannot determine if MLLMs can handle real-world tasks.

Improving MLLM performance for this setting also remains insufficiently studied. Existing approaches typically train models to align chart images directly with full tables [47, 52, 78], but we find that such end-to-end mappings struggle to recover accurate values when visible data labels are absent. We introduce a human-inspired training framework that strengthens coordinate geometry perception before learning full table extraction. This progressive strategy yields state-of-the-art accuracy on chart data extraction and provides a more reliable foundation for mixed-initiative workflows.

3 Background

In this section, we first present a motivating scenario that illustrates the challenges with current tools. We then report interviews with practitioners to understand their needs and perceptions of MLLM reliability in chart data extraction.

3.1 Motivating Scenario

To give context to our study, we present a motivating scenario involving a researcher named Alex, who aims to analyze the relationship between review scores and acceptance rates at an academic conference. Alex has collected several charts from annual conference reports, but does not have access to the underlying datasets. To carry out the analysis, Alex must extract the numerical values encoded in these charts.

Chart Characteristics. All charts are stacked bar charts. The x-axis denotes the review scores assigned to manuscripts, the y-axis shows the number of submissions, and the stacked segments represent accepted versus rejected papers. Although the charts share a similar semantic structure, their color palettes and layouts vary across years due to changes in conference branding. Importantly, none of the charts include data labels on the stacked segments.

Trying WebPlotDigitizer. Because the charts have no data labels, Alex rules out OCR-based methods and begins with WebPlotDigitizer, a widely used interactive extraction tool. Alex initially tries the color-detection mode but finds the results too noisy to be usable. He switches to manual annotation, where he must calibrate axes and click each bar segment individually. The process is tedious and slow. Moreover, the extracted table requires additional post-processing: score ranges are not included in the output, and when clicking the upper segment of a stacked bar, the tool returns the cumulative height rather than the segment-specific value. As Alex remarks, *"It will take me hours to finish extracting data from all these charts, and then I still have to write scripts to clean the tables. It is too much work."*

Exploring Mixed-Initiative Tools. Alex then looks for more automatic solutions and discovers mixed-initiative systems, which combine automation with user control. However, he cannot find any tools that support his needs. For example, he finds ChartDetective appears promising, but it requires vector graphics as input, and he only has raster images. ChartSense, another system designed by HCI researchers, does not support stacked bar charts. Alex realizes that these systems are not general enough for his needs.

ChatGPT to the Rescue? Still searching for a better option, Alex turns to ChatGPT with multimodal capabilities. *"It can probably give me some data, but I do not know how accurate it will be,"* he thinks. He uploads a chart and prompts GPT-4o: *"Extract the data from this chart and output a table."* The model produces a table with the correct structure, but after a quick visual comparison, Alex notices several values that are clearly incorrect. *"If even I can spot these errors at a glance, how can I trust the rest?"* Alex wonders. With no reliable automatic solution, Alex ultimately returns to WebPlotDigitizer, accepting the effort required. *"I really hope there is a better way to do this,"* he sighs.

Summary. This scenario illustrates common challenges in real-world chart data extraction. Interactive tools, although reliable, are tedious and time-consuming. Mixed-initiative systems offer partial automation but often lack support for diverse chart types or generalize poorly. MLLMs present an appealing alternative due to their flexible, instruction-following nature, yet their reliability remains uncertain, especially when charts lack data labels.

3.2 Interview with Practitioners

To better understand how practitioners perceive MLLMs for chart data extraction, particularly their reliability, we then conducted interviews with five experienced practitioners (denoted as P1—P5; ages 23—28), each of whom had performed chart data extraction more than ten times in their professional workflows. Their backgrounds span finance, STEM research, and social science, offering a diverse set of perspectives.

3.2.1 Procedure. We conducted semi-structured interviews that encouraged participants to reflect on their past extraction workflows and to evaluate the reliability of different tools they had used. We focus on two main themes: (1) the tools they currently use, as well as the efficiency, accuracy, and usability of these tools; (2) their impressions and experiences, if any, with using MLLMs for chart data extraction. All interviews were recorded and transcribed. We summarized the transcripts and identified recurring themes. The analysis focuses on synthesizing practitioners' perspectives on the reliability of the tools they use and MLLMs.

3.2.2 Feedback. **Interactive tools are reliable but tedious.** All participants use interactive tools such as WebPlotDigitizer or Origin as their primary choice. The dominant justification was reliability: these tools offer a transparent, verifiable workflow in which each extracted value can be visually anchored to a specific mark. P5, a biomedical researcher, explained: *"I trust it because I can see exactly where each annotated point is in the chart. It is intuitive and ensures pixel-level accuracy."* Mark annotations were consistently described as the most time-consuming steps, especially when processing thousands of data points. All participants expressed a strong interest in more efficient solutions.

MLLMs are convenient but not accurate. Four out of five participants had experimented with MLLMs such as GPT-4 or Gemini by uploading chart screenshots and requesting tables or re-rendering code, except for P5. He explained: *"I once uploaded a chart and asked for a value, but the result was inaccurate, so I never tried to extract the full table."* They appreciated the convenience of natural language prompting, but reliability was uniformly judged as low due to two issues:

(1) Numerical inaccuracies. Participants who tried MLLMs consistently reported encountering incorrect values in the outputs. P3 noted a common failure pattern: *"When I ask the model to extract the entire table from a chart, it often repeats a value in several consecutive cells instead of the actual value."*
(2) Lack of visual grounding. Participants can easily check if there is missing data in the output, but cannot verify whether the predicted values are accurate. P2 remarked: *"If the model were accurate, this would not matter. But right now, I cannot confirm whether the values are correct."*

Reliability is non-negotiable. Participants emphasized that reliability determines whether a tool can be adopted in practice. The reasons fall into two categories:

(1) Research rigor. Currently, MLLMs are not proven reliable for data extraction in research contexts. P1, a graduate student who extracted data from charts in scientific papers for meta-analyses, noted: *"We need to report how data is obtained. If the extraction method is not considered reliable, the results will be questioned."*
(2) Downstream risk. In many applications, extracted values are used for further analysis or decision-making, and inaccuracies can lead to harmful decisions. P4, a financial analyst, noted: *"A small error can lead to the wrong investment decision."* P3 added: *"We extract historical chart data to train disease progression models. The data should be accurate for the model to learn correctly."*

Summary. Despite the potential of MLLMs for efficient, unified chart data extraction, they are not perceived as reliable. Therefore, before MLLMs can be integrated into practical workflows, two challenges must be addressed: First, quantifying their unreliability in chart data extraction, which can guide future improvements and serve as proof for model reliability; second, developing techniques to improve MLLMs' accuracy and support reliable use.

4 Preliminary Study

To quantify the reliability of MLLMs in chart data extraction, a benchmark must systematically evaluate their capabilities and limitations. To guide the design of such a benchmark, we first decompose the task and conduct two small-scale pilot experiments to explore factors that influence model performance.

Task Definition. Chart data extraction refers to recovering the numerical values encoded in a chart image. Given a chart image I, the goal is to produce a table T that accurately reflects the values represented by the chart's visual marks. In most real-world scenarios, extracted tables are used for further analysis, so a useful extraction should satisfy two criteria: (1) *Correct Structure:* The table must contain the correct number of rows and columns with appropriate headers to remain compatible with downstream processing. (2) *Accurate Values:* Each numerical entry should closely match the values encoded in the chart to maintain analytical validity. These criteria reflect two levels of understanding: structure recovery tests whether the model can parse the chart layout, while value accuracy tests whether the model can map visual encodings to precise numbers. They also differ in visibility. In practice, users can easily spot errors in the table structure, but errors in the numerical values are much harder to detect. Therefore, we design two pilot experiments to separately study each aspect.

4.1 RQ1: Can MLLMs Generate Correct Table Structure?

To answer RQ1, we evaluate the ability of MLLMs to identify rows, columns, and headers without introducing, omitting, or misinterpreting table elements.

Experiment Setup. We randomly selected 200 charts from ChartQA [48]: 100 "two-column" charts (basic bar, basic line, pie) and 100 "multi-column" charts (grouped bar, stacked bar, grouped line), covering a range of chart types and different table structures. For this experiment, we used Gemini 2.5 Flash [12], an advanced MLLM. We chose Gemini 2.5 Flash due to its strong performance in recent VQA benchmarks, making it a representative candidate for evaluating the current capabilities of high-performing MLLMs.

Each chart was shown to the model with the prompt: *"Extract the data from this chart and output it in CSV format."* The generated tables were then manually evaluated against the ground truth. We categorized the outputs into four groups: "correct structure," "missing rows/columns," "redundant rows/columns," and "misinterpreted row/column ."

Table 1: Gemini 2.5 Flash's performance on generating the correct table structure.

Correct	Missing	Redundant	Misinterpreted
194/200 (97.0%)	0	4/200 (2.0%)	2/200 (1.0%)

Results. As Table 1 shows, Gemini 2.5 Flash successfully generated the correct table structure for 97% of charts, showing that advanced MLLMs can recognize overall chart layouts and produce well-formed tabular formats. This suggests that identifying rows, columns, and headers is largely a solved problem. The major remaining challenge lies in accurately recovering the numerical values that populate these structures.

4.2 RQ2: Can MLLMs Recover Accurate Values?

To answer RQ2, we evaluate how value accuracy is affected by two factors: (1) the presence or absence of data labels, and (2) the prompting strategy (single value at a time vs. full table extraction).

We aim to answer the following question: (1) To what extent do data labels affect numerical accuracy? (2) Does a prompting strategy that produces longer outputs to extract all values at once cause lower accuracy?

Data Preparation. We randomly sampled 50 Vega-Lite [59] specifications from nvBench [45], with 10 instances each for basic bar, stacked bar, basic line, grouped line, and pie charts, ensuring diversity in chart types and table structure complexities. For each Vega-Lite specification, we edited it with Python scripts and rendered two chart versions: one with data labels and one without, resulting in 100 charts (50 with data labels and 50 without).

Experiment Setup. We again used Gemini 2.5 Flash [12] for this experiment. Two prompting strategies were tested:

- **Single-value prompt:** *"What is the value of [X]?"*, where [X] refers to a specific data point. Each value was queried individually, reducing reasoning effort but requiring multiple prompts, which is token-inefficient.
- **Full-table prompt:** *"Extract the data from this chart and output it in CSV format."* We also provided the model with the corresponding CSV, in which the values are masked for the model to fill in. Prompting for a table is far more efficient for real-world use, but it needs models to handle longer output while maintaining accuracy across all values.

Applying both prompts to all 100 charts produced 200 tables.

Evaluation Metric. We computed numerical accuracy using MAPE:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{v_i - \hat{v}_i}{v_i}\right| \times 100\%, \tag{1}$$

where v_i is the ground truth value, $\hat{v}_i$ is the predicted value, and n is the total number of values. We exclude zero values, and cap individual errors at 100% to mitigate the influence of extreme outliers.

Table 2: Gemini 2.5 Flash's accuracy in generating data values under two pairs of conditions: charts with vs. without data labels, and single-value prompt vs. full-table prompt. The values are reported in MAPE. Lower is better.

Prompt Strategy	w/ Data Labels	w/o Data Labels
single value at a time	1.8%	7.4%
full table prompting	1.3%	7.2%

Results. Table 2 shows clear differences across conditions. The presence of data labels substantially improves value accuracy. When labels are visible, the model achieves an MAPE of less than 2%, considered good. In contrast, without data labels, the MAPE increases

to over 7%, which is high enough to undermine the reliability required for analytical tasks. The choice of prompting strategy has only a minor effect: both the single-value prompt and the full-table prompt yield similar MAPE, with the full-table prompt even slightly better. This suggests that while full table prompting is more efficient in practice without accuracy trade-offs, the central bottleneck lies in the model's ability to map visual encodings to precise numbers.

4.3 Summary

This preliminary study offers several insights for addressing the challenges of chart data extraction with MLLMs:

Value accuracy is the main bottleneck. MLLMs are good at reconstructing table structure but fail at recovering accurate values, especially when data labels are absent. This gap shows that, when we evaluate and improve MLLMs for chart data extraction, numerical accuracy should be the primary focus.

Unlabeled charts are the critical test case. Removing data labels increases MAPE by roughly 5%, which is a disaster for practical use. Therefore, when evaluating the MLLMs, we need to eliminate the effect of data labels and focus on the unlabeled chart, which reflects real-world needs.

Full table prompting is a good practice. Our results show that full table prompting is efficient, token-economical, and does not reduce accuracy compared with single-value prompting. Thus, full table prompting can be a practical default for chart data extraction.

5 ExChart-Bench

Our preliminary study showed that although MLLMs can reliably generate correct table structures, their practical adoption is constrained by value inaccuracy, especially when charts lack data labels. Practitioners emphasized their concerns about numerical fidelity, and our pilot experiments confirmed that value accuracy remains the primary bottleneck for current MLLMs. These findings highlight the need for a systematic benchmark that can reveal such limitations and guide future development.

A dedicated benchmark serves two key purposes. First, it provides a standardized framework to assess and compare different models, offering evidence of their reliability in chart data extraction. Second, it drives progress by surfacing performance bottlenecks and informing the design of more accurate and robust methods.

5.1 Task Refinement

To more effectively assess the core capability of MLLMs in chart data extraction, we refine the task with a focused formulation.

Given a chart image I without data labels and a prompt P_{temp} containing a corresponding CSV template with placeholders (as illustrated in Figure 2), where numerical values are masked but headers and non-value cells are provided, the objective is to generate a completed table T by filling in the placeholders with accurate numerical values. This refined task isolates the model's ability to interpret visual encodings and infer precise values, an essential requirement in practical chart data extraction and the central bottleneck identified in previous studies.

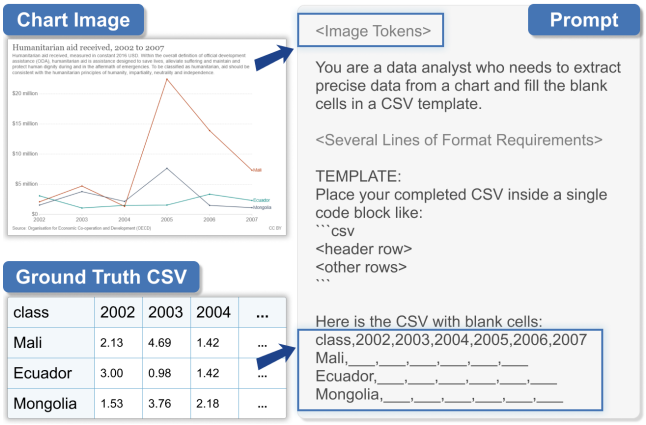

Figure 2: Prompt creation in ExChart-Bench. We provide the model with a CSV template that corresponds to the chart. In the template, values are marked with placeholders.

5.2 Dataset Construction Pipeline

Based on the refined task definition, we then construct the ExChart-Bench. Each sample in the final dataset includes a chart image without data labels, a ground truth CSV data table, and a prompt containing the corresponding CSV template. Figure 3 illustrates the overall construction pipeline, which consists of the following steps:

Source Preparation. We collect both real-world and synthetic charts to ensure diversity across chart types and visual complexities. For real-world charts, we source images from publicly available datasets such as ChartQAPro [46] and CharXiv [67], as well as additional charts crawled from the web. For synthetic charts, we gathered Vega-Lite specifications from existing collections, the online Vega-Lite gallery [34], and Python scripts from prior datasets. We also include charts from datasets that provide synthetic images with associated JSON metadata [7, 79]. After collecting a large set of sources, we filter out unusable samples. We remove scripts and specifications that cannot be correctly rendered. For image-metadata pairs, we use Qwen2.5-VL 32B to identify samples with missing metadata or visible data labels and filter them out.

Diversity Expansion. To further broaden coverage, we generate additional charts derived from existing Matplotlib and Vega-Lite specifications. For style diversity, we randomly vary supported attributes such as color palettes, fonts, gridline visibility, and tick formatting. For data diversity, we apply transformations including scaling, shifting, and adding noise to the underlying datasets. We also modify chart configurations, for example, converting grouped bar charts into stacked versions, to enrich structural variety.

Data Label Removal. For real-world charts, we manually remove all visible data labels using image editing tools. We first mask the labels with background color and then carefully recover occluded regions by cloning surrounding pixels. For synthetic sources, such as Python scripts or Vega/Vega-Lite specifications, we write Python scripts to remove label-related configurations.

Synthetic Chart Augmentation. To simulate imperfections commonly observed in scanned or low-quality images, we apply several augmentation techniques to rendered synthetic charts. These

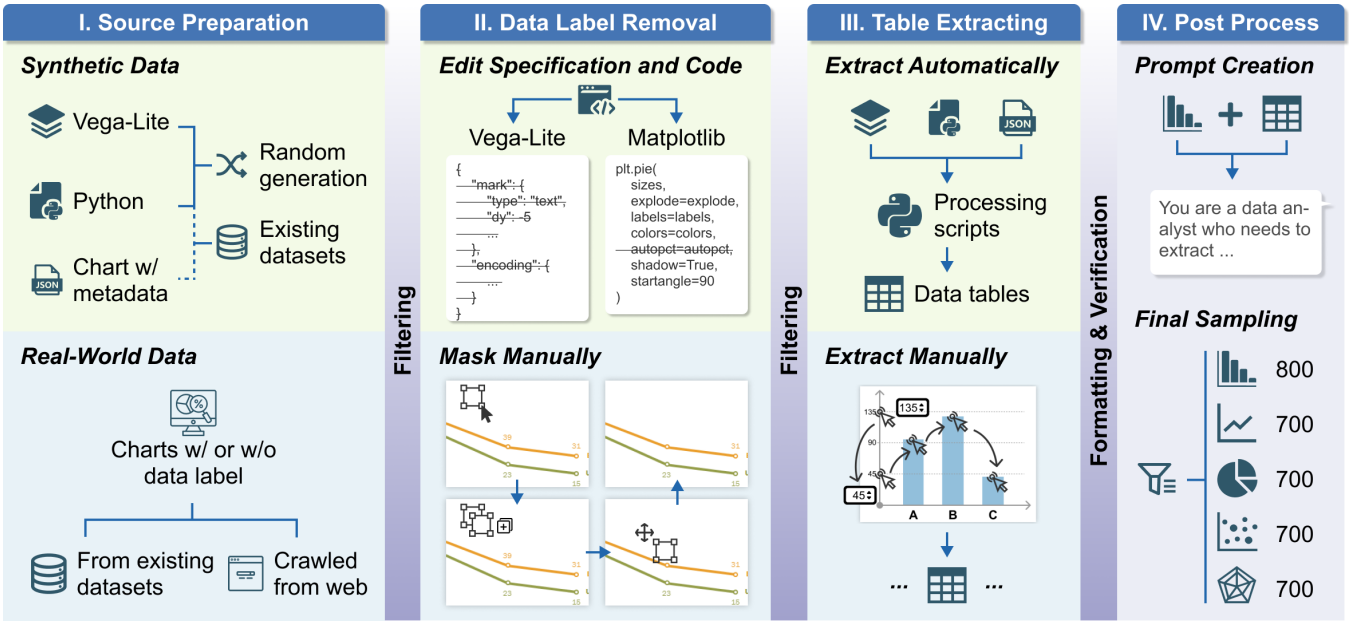

Figure 3: The benchmark dataset construction pipeline. We collected both real-world and synthetic chart sources, removed data labels, extracted data tables, and created prompts, resulting in a dataset of chart-CSV-prompt triplets.

include small random rotations (within ±5°), Gaussian noise, sharpening, contrast adjustments, ink-saving effects, and brightness variation. We also apply downscaling to mimic low-resolution charts and use grayscale conversion or color inversion to simulate monochrome or unconventional color schemes. These augmentations are randomly applied to all synthetic charts.

Data Table Extraction. To obtain the ground truth CSV tables, we used different approaches for the synthetic and real-world charts. For real-world charts, we manually annotate data points using WebPlotDigitizer [43] for maximum precision. For synthetic charts, we write Python scripts to extract data directly from original scripts, specifications, or metadata. We use Qwen2.5-VL 32B to check for consistency between the extracted tables and the charts and correct any discrepancies manually.

Table Formatting and Verification. We standardize table formats within each chart type to ensure structural consistency. For each chart type, we select a canonical format for each chart type that results in a minimal number of cells to save tokens, and we convert all tables accordingly while preserving all information. Unit usage, delimiter styles, and column ordering are harmonized automatically using Python scripts.

Prompt Creation. We design a prompt template as shown in Figure 2, which clearly defines the task and provides the model with structured input. Each prompt includes an instruction followed by a CSV template that corresponds to the chart. In this template, all numeric values are replaced with placeholders, and the headers and structure are retained. We also instruct the model to output its answer in a code block for easier parsing.

Final Sampling. To avoid unresolvable ambiguities, we remove charts with excessive mark overlap, which makes estimating values infeasible. Then, we sample a balanced subset of charts across types, ensuring diversity in both type and style in the final benchmark. Since bar charts comprise a significantly larger proportion of the collected data and have more variants (e.g., stacked, grouped, vertical, and horizontal), we sample 800 bar charts and 700 charts from each of the other categories.

Table 3: ExChart-Bench overview.

Total	By Chart Type					By Source	
	Bar	Line	Scatter	Pie	Radar	Real	Synthetic
Chart Number							
3,600	800	700	700	700	700	744	2,856
Value Number							
33,757	9,500	9,494	8,414	3,055	3,294	9,907	23,850

5.3 Benchmark Overview

ExChart-Bench contains 3,600 chart-table-prompt pairs spanning five common chart types: bar (basic, grouped, stacked), line (basic, grouped), scatter, pie, and radar charts. Among these, 744 are real-world charts, and 2,856 are synthetic charts. In total, the dataset includes 33,757 numerical values. Table 3 summarizes the dataset statistics. To support the analysis of model performance with respect to table size, we categorize the charts into two groups: smaller and larger tables. For each chart type, we sort the charts by number of values and split them to balance the total number of values in the "smaller" and "larger" subsets.

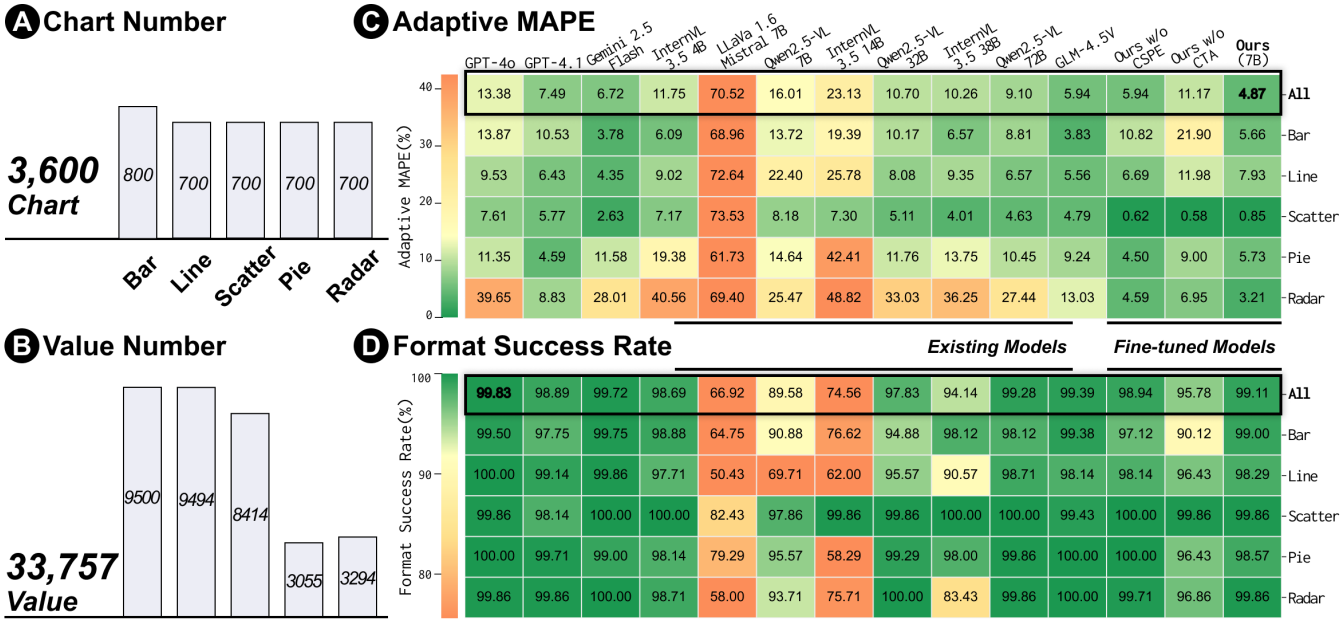

Figure 4: Distribution of benchmark samples and evaluation results of MLLMs across different chart types. (A) Distribution of chart types in the benchmark. (B) Distribution of numerical values in the benchmark. (C&D) Adaptive MAPE and format success rate of various MLLMs on the benchmark.

5.4 Evaluation Metrics

To evaluate model outputs, we first parse the generated text into CSV tables. Since the models are instructed to output tables within code blocks, we begin by extracting the content enclosed by the code block fences. The extracted text is then parsed as a serialized CSV using Python scripts. For each value cell, we apply regular expressions to extract the numerical values, discarding any additional text or formatting.

Format Success Rate. If a model fails to produce a parsable CSV or outputs a table whose shape does not match the provided CSV template, we consider it a format failure. This metric reflects the model's ability to follow output formatting instructions. Even with a template provided, some models still produce incorrect table shapes, suggesting that without such guidance, their ability to output a correct data table would be even more limited.

Adaptive Mean Absolute Percentage Error. To evaluate numerical accuracy, we define Adaptive Mean Absolute Percentage Error (Adaptive MAPE), a variant of MAPE designed to better reflect perceptual error in chart data extraction tasks. Unlike standard MAPE or sMAPE, which divide by the ground truth or average of predicted and ground truth values, Adaptive MAPE divides the absolute error by the maximum absolute value in the chart. This adjustment reduces the disproportionate impact of small values on the overall error. For instance, predicting 0.1 instead of 0.001 in a chart ranging from 0 to 100 results in a large MAPE, despite being visually indistinguishable.

Formally, for each chart, let $V_{\max}$ denote the maximum absolute ground truth value. Given predicted values $\hat{v}_i$ and ground truth values v_i over n cells, Adaptive MAPE is defined as:

$$\text{Adaptive MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{v}_i - v_i}{V_{\max}}\right| \times 100\% \tag{2}$$

For any individual error above 100%, we cap it at 100% to mitigate the effect of extreme outliers. Unparsable values (e.g., non-numeric text or empty cells) are treated as having 100% error. For a fair comparison, values from unparsable output or shape mismatch tables are also treated as having a 100% error. By normalizing to the global scale of the chart rather than local value magnitudes, Adaptive MAPE provides a more perceptually meaningful and robust measure of accuracy.

5.5 Overall Analysis

We evaluate a wide range of advanced MLLMs, including commercial proprietary models such as GPT-4o [26], GPT-4.1 [27], and Gemini 2.5 Flash [12], as well as open-source models of various scales, including the LLaVA-Next [38–40], Qwen [2], GLM [18], and InternVL [11] families. The performance across different chart types and table sizes is summarized in Table 4 and Table 5, respectively.

Our analysis reveals several key findings:

F1: None of these MLLMs are ready for chart data extraction. Even the best-performing model, GLM-4.5V, achieves a 99.39% format success rate but only a 5.94% Adaptive MAPE. Since we are using Adaptive MAPE rather than MAPE, the real percentage error for each value would be higher than 5.94%. This level of error can be unacceptable in domains where precision is critical.

F2: Scaling up MLLMs on general VQA or CQA is not sufficient. In terms of Adaptive MAPE, GLM-4.5V, which has more

Table 4: Model performance across different chart types. Format Success Rate measures the ability to follow instructions and output correct table structures; the higher the better. Adaptive MAPE measures overall accuracy; the lower the better.

Model	Format Success Rate (%) ↑						Adaptive MAPE (%) ↓					
	All	Bar	Line	Scatter	Pie	Radar	All	Bar	Line	Scatter	Pie	Radar
Commercial proprietary models												
GPT-4o	**99.83**	99.50	100.00	99.86	100.00	99.86	13.38	13.87	9.53	7.61	11.35	39.65
GPT-4.1	98.89	97.75	99.14	98.14	99.71	99.86	7.49	10.53	6.43	5.77	4.59	8.83
Gemini 2.5 Flash	99.72	99.75	99.86	100.00	99.00	100.00	6.72	3.78	4.35	2.63	11.58	28.01
Open-source models (ordered by model size)												
InternVL3.5 4B	98.69	98.88	97.71	100.00	98.14	98.71	11.75	6.09	9.02	7.17	19.38	40.56
LLaVA 1.6 Mistral 7B	66.92	64.75	50.43	82.43	79.29	58.00	70.52	68.96	72.64	73.53	61.73	69.40
Qwen2.5-VL 7B	89.58	90.88	69.71	97.86	95.57	93.71	16.01	13.72	22.40	8.18	14.64	25.47
Ours w/o CSPE (7B)	98.94	97.12	98.14	100.00	100.00	99.71	5.94	10.82	6.69	0.62	4.50	4.59
Ours w/o CTA (7B)	95.78	90.12	96.43	99.86	96.43	96.86	11.17	21.90	11.98	0.58	9.00	6.95
Ours (7B)	99.11	99.00	98.29	99.86	98.57	99.86	**4.87**	5.66	7.93	0.85	5.73	3.21
InternVL3.5 14B	74.56	76.62	62.00	99.86	58.29	75.71	23.13	19.39	25.78	7.30	42.41	48.82
Qwen2.5-VL 32B	97.83	94.88	95.57	99.86	99.29	100.00	10.70	10.17	8.08	5.11	11.76	33.03
InternVL3.5 38B	94.14	98.12	90.57	100.00	98.00	83.43	10.26	6.57	9.35	4.01	13.75	36.25
Qwen2.5-VL 72B	99.28	98.12	98.71	100.00	99.86	99.86	9.10	8.81	6.57	4.63	10.45	27.44
GLM-4.5V (>100B)	99.39	99.38	98.14	99.43	100.00	100.00	5.94	3.83	5.56	4.79	9.24	13.03

Table 5: Impact of table size on model performance. Entries highlighted in green indicate the better result in each pair; metrics mirror Table 4.

Model	Format Success Rate (%) ↑		Adaptive MAPE (%) ↓	
	Smaller	Larger	Smaller	Larger
Commercial proprietary models				
GPT-4o	99.83	99.84	12.60	14.16
GPT-4.1	98.93	98.82	6.37	8.60
Gemini 2.5 Flash	99.83	99.53	6.44	7.01
Open-source models				
Qwen2.5-VL 7B	86.29	95.60	18.74	13.29
Qwen2.5-VL 32B	97.38	98.67	10.48	10.91
Qwen2.5-VL 72B	99.57	98.74	6.90	11.30
InternVL3.5 4B	99.05	98.04	10.35	13.14
InternVL3.5 14B	65.91	90.35	30.08	16.20
InternVL3.5 38B	94.45	93.56	9.57	10.95
LLaVA 1.6 Mistral 7B	64.66	71.04	67.73	73.30
GLM-4.5V	99.53	99.14	5.04	6.84

than 100B parameters in total, leads all models. Commercial proprietary models like Gemini 2.5 Flash, GPT-4.1, also have comparative performance. This is intuitive as these models have been trained on large-scale data and optimized for a wide range of tasks. However, there are also some exceptions. GPT-4o, despite being a large and powerful model, performs worse than InternVL3.5 38B and Qwen2.5-VL 72B. Smaller models like InternVL3.5 4B also outperform larger ones like InternVL3.5 14B and Qwen2.5-VL 7B. It is interesting that in the same model family, the performance does not always improve with scale. This suggests that scaling up general VQA or CQA training does not necessarily improve the capability to extract chart data. We may not need very large models for chart data extraction.

F3: Tailored data and training methods matter more. We found that general VQA benchmarks are not reliable indicators of performance on chart data extraction. The rankings of MLLMs on benchmarks such as MMBench [42] do not align well with their performance on our task. This discrepancy underscores the need for specialized evaluation: chart data extraction requires precise geometric reasoning and fine-grained value estimation, which are capabilities that are not fully exercised in standard VQA or even typical CQA tasks. Combined with **F2**, this suggests that for chart data extraction, task-specific data and training strategies may offer a more cost-effective path to improving model performance.

F4: Even smaller models can output correct table structure. 7/11 models achieve over 97% format success rate, indicating that they can understand and follow the output format requirements well. This aligns with our preliminary study and meets our expectations. By providing a CSV template in the prompt, we guide the model to generate the desired structure. This capability is crucial for practical deployment, where strict adherence to table structure is often necessary. This also inspires us that, in future applications, users can provide a table to be filled with MLLMs for more controllable data extraction.

F5: Significant imbalance exists in accuracy across chart types. As shown in Figure 4, for most MLLMs, bar charts, line charts, and scatter plots, which use Cartesian coordinates and are commonly seen in training corpora, result in higher accuracy. In contrast, radar charts, which use polar coordinates and are relatively rare, remain the most difficult. Interestingly, GPT-4.1 achieves the best score on pie and radar charts (4.59% and 8.83%, respectively) but performs worse on bar charts. Such an imbalance may stem from the training data distribution, multi-task instruction tuning, or distillation processes.

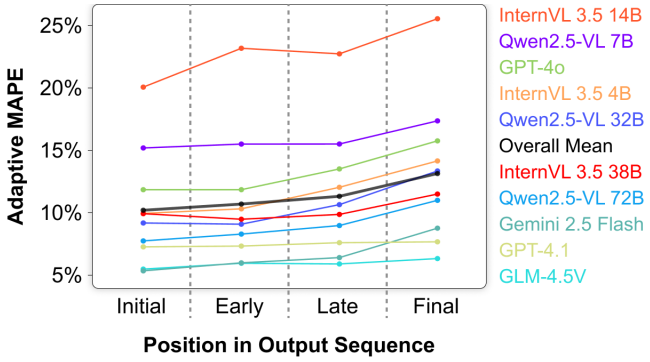

Figure 5: Trends in Adaptive MAPE relative to the position of values in the output sequence. "Initial" refers to the first 25% of values in sequences, "early" to the next 25%, "late" to the following 25%, and "final" to the last 25%.

F6: Table size has a limited impact on overall performance, but value accuracy degrades later in the output sequence. As shown in Table 5, format success rates are comparable in general across different table sizes, with minor variations between models. For Adaptive MAPE, some models perform slightly better on smaller tables, while others show improved accuracy on larger ones. Overall, smaller tables show a marginal advantage, but the effect is not substantial, suggesting that table size alone is not a strong determinant of performance. However, when analyzing the Adaptive MAPE in relation to the position within the output sequence (Figure 5), a consistent trend emerges: values generated later in the sequence tend to have higher error. This indicates that while table size may not directly impact performance, the sequential nature of autoregressive text generation contributes to error accumulation, making later values less reliable. Models may partially rely on earlier output values and the geometry relationships between corresponding visual marks to estimate subsequent values. Such dependency can be both beneficial and harmful. Further exploration can be considered to mitigate this issue and improve robustness.

5.6 Analysis of Failure and Successful Cases

We analyzed the typical failure modes across models and identified several recurring patterns:

Sudden changes in value trends are often misinterpreted in line charts. As shown in Figure 6 (A), line charts with sharp spikes or drops frequently lead to inaccurate predictions. Models tend to repeat the same values or assume smoother trends, likely reflecting prior expectations learned during pretraining. This indicates a lack of robustness to abrupt visual transitions.

Models become less accurate when the slice in pie charts exceeds 50%. As shown in Figure 6 (B), compared to a small slice, when a pie slice represents more than half of the chart, models are more likely to produce larger errors in estimating its value. This may be due to a minority of large slices in training data, or the visual challenge of accurately judging angles and areas in such cases.

Small values near the origin are overestimated in radar charts. As illustrated in Figure 6 (C), values within the innermost grid rings, which represent the smallest magnitudes, are frequently inflated. This may stem from challenges in distinguishing fine-grained radial distances near the center, particularly in polar coordinate systems.

Zero-height bars often cause corruption in outputs. As shown in Figure 6 (D), when a bar chart includes bars with zero height, models frequently mess up the corresponding and following values. This phenomenon is more obvious in stacked bar charts and grouped bar charts, as models tend to interpolate or replicate nearby values. This suggests that models may struggle with discontinuities in visual encodings. Training on more examples with zero values may help models better handle such cases.

Models are not good at counting the number of digits in large-magnitude values. As shown in Figure 6 (E), when axis ticks have large magnitudes (e.g., $> 10^5$), models frequently miscount the number of digits, leading to errors in magnitude. This is consistent with known weaknesses in LLMs regarding symbolic precision, similar to errors in character-counting tasks (e.g., "how many r's in strawberry").

Beyond the representative failure cases, we also present success examples in Figure 7 to illustrate the upper bound of current MLLM capabilities. This contrast underscores the necessity of targeted training and interactive validation to address the long tail of challenging real-world charts.

6 ExChart

Evaluation results on ExChart-Bench indicate that existing MLLMs struggle with chart data extraction, particularly in achieving high numerical accuracy. While these models perform well on general vision-language tasks, they often lack the specialized capabilities needed to interpret chart-specific visual encodings.

Motivated by insights from visualization and HCI research on how people interpret or generate charts [22, 23, 61], we introduce ExChart, a two-stage training framework that mirrors the progressive learning process. Rather than directly aligning full charts with tables or code, we decompose the task into foundational perception and higher-level extraction. The first stage, *Coordinate System Perception Enhancement* (CSPE), is designed to improve the model's understanding of coordinate geometry and visual encodings. The second stage, *Chart-Table Alignment* (CTA), fine-tunes the model to generate structured data tables with high numerical fidelity. An overview of the full framework is presented in Figure 8.

6.1 Coordinate System Perception Enhancement

This stage aims to improve the model's understanding of coordinate geometry, enabling it to estimate the chart coordinates of individual visual marks. **F5** reveals an imbalanced performance across chart types: most MLLMs perform better on charts using Cartesian coordinates (e.g., bar and line charts) than on those using polar coordinates (e.g., pie and radar charts).

Intuitively, before a model can reliably recover data values from a chart, it must first grasp the coordinate system that governs the visual encodings. This reflects how humans learn to interpret charts, first understanding coordinate systems and then mapping visual cues to values. To facilitate this process, we design a pretraining task that explicitly targets coordinate system understanding. Additionally, our analysis shows that models may rely on visual relationships

Figure 6: Common failure cases of MLLMs on our benchmark: (A) Sudden changes in value on line charts. (B) Pie slices exceeding 50%. (C) Small values in radar charts. (D) Zero values in bar charts. (E) Large-magnitude axis ticks.

Figure 7: Success examples, where each chart achieves less than 1% average Adaptive MAPE across Gemini 2.5 Flash, GLM-4.5V, and GPT-4.1.

between marks to infer values, leading to error accumulation in sequential outputs. By explicitly training models to reason about coordinate geometry, we aim to reduce this dependency and improve overall robustness.

In this task, for each type of coordinate system, the model is given images containing a single visual mark and must predict its coordinates. For Cartesian systems, the prediction takes the form of (x, y); for polar systems, it is expressed as (r, θ). This task serves as an atomic subcomponent of chart data extraction, isolating the challenge of coordinate interpretation from the complexities of full-table generation.

Training Objective. We formulate this task as a text generation problem and optimize it using cross-entropy loss. In this stage, the output could be just the coordinate pair, or it could also be output in CSV format. The example of Cartesian coordinates output in both formats is shown below:

Output Format 1:

```
(23.5, 47.8)
```

Output Format 2:

```
```csv
x,y
23.5,47.8
```
```

To explore the impact of output format on this stage, we experiment with both formats in two-stage training and ablation studies and adopt the better one.

Validation Metric. We evaluate performance using MAPE, treating each numeric value in the coordinate pair as an individual data point.

Data Generation. We construct a large-scale synthetic dataset that includes both Cartesian and polar coordinate systems, consistent with the types of charts used in our benchmark. For Cartesian coordinates, we use the Vega specifications to generate images, each containing a single point. The Axis ranges are randomly sampled to span a wide range of magnitudes, and point positions are randomly selected within these ranges. To promote style diversity, we vary Vega styling options such as axis placement, tick mark appearance, background color, point shape and size, and canvas resolution.

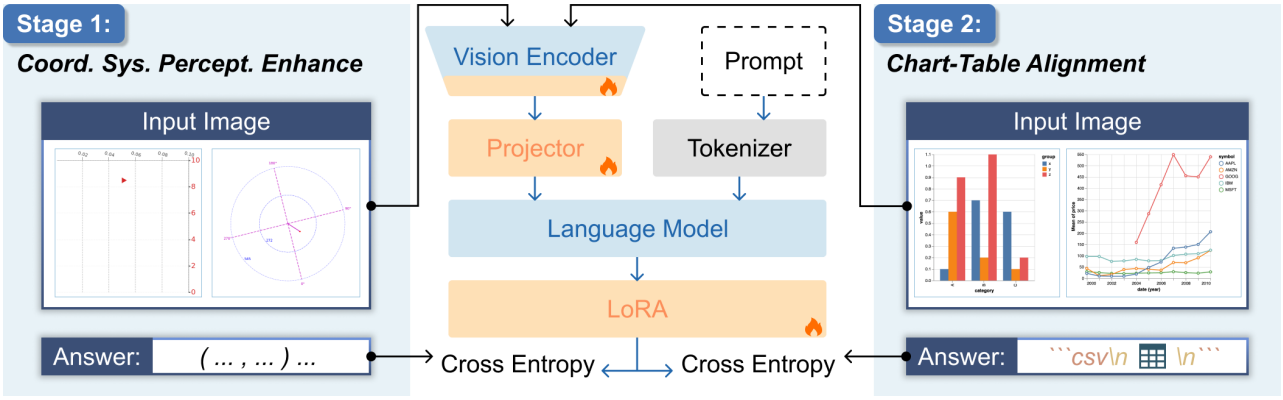

Figure 8: The proposed training framework includes a *Coordinate System Perception Enhancement* stage and a *Chart-Table Alignment* stage. During training, the final four layers of the vision encoder and the multimodal projector are fully fine-tuned, while the language model is adapted using LoRA.

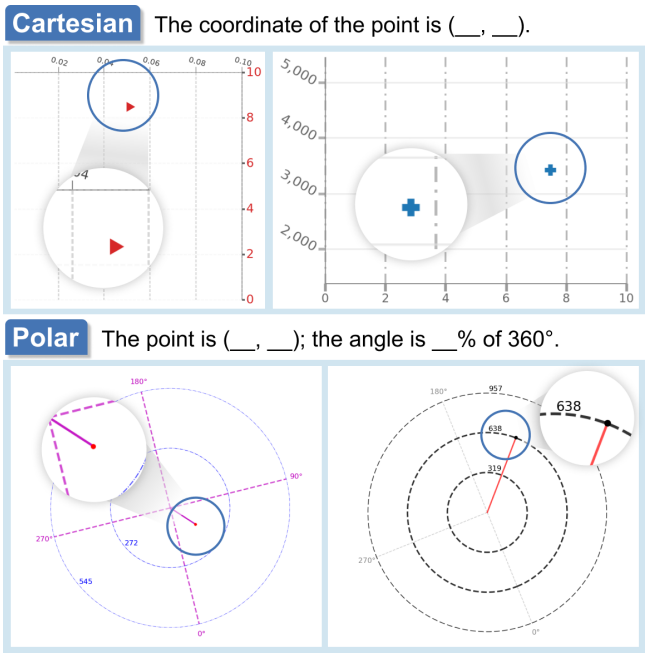

Figure 9: Examples of training samples for the Coordinate System Perception Enhancement stage.

For polar coordinates, we use Matplotlib to generate single-point charts. We vary the reference direction (i.e., the 0° axis) and the radius range. Points are placed at random angles between 0° and 360°, with radii randomly sampled from the specified range. Chart appearance is further diversified using Matplotlib's style parameters. To simulate pie chart semantics, we additionally include the corresponding percentage (based on the angle) in the model output, ask the model to output r, θ, and percentage.

This generation pipeline allows us to efficiently produce 40,000 samples for each type of coordinate system, and 1,000 samples for validation. For all samples, we also apply the same augmenting strategy as in the benchmark to simulate imperfection in real-world charts. This pipeline can also be easily extended to include additional coordinate systems or visual styles as needed. Examples of samples are illustrated in Figure 9.

6.2 Chart-Table Alignment

As with many vision-language tasks, fine-tuning on task-specific data is crucial for aligning the model with the intended objective. In our case, we fine-tune the model on chart-table pairs to improve its ability to extract structured data from visual representations. This stage builds upon the first: once the model has learned to interpret coordinate systems, it is further trained to recognize visual marks and translate them into accurate, structured table outputs.

Training Objective. We optimize the model using cross-entropy loss as well. Given a chart and an instruction prompt, the model generates a predicted table in plain text. The training loss is computed between the predicted table and the ground truth table.

Validation Metric. For validation, we use the same prompt template as defined in our benchmark, as shown in Figure 2. After parsing the model's output into a structured table, we calculate the Adaptive MAPE between the predicted and ground-truth values. To ensure robust evaluation, individual errors are capped at 100%, and any unparsable or missing values are also assigned a 100% error.

Data Generation. We employ the same generation pipeline as used in the synthetic portion of our benchmark to create chart-table pairs. Due to the limited availability of high-quality open-source chart datasets, we primarily rely on synthetic data generated using Vega and Matplotlib. We generate a total of 50,000 samples for training and 1000 samples for validation, covering a wide range of chart types, styles, and data distributions.

In this stage, the prompt used in training is slightly different from that used in validation. Instead of providing a partially completed CSV template to fill in, during training, the prompt instructs the model to generate the entire table from scratch within a code block.

6.3 Model Configuration

Base Model. We use Qwen2.5-VL 7B as the base model for all experiments. This model is widely adopted for vision-language tasks and has demonstrated strong performance across a variety of VQA and CQA benchmarks.

Training Setup. Our goal is to train the model to map visual elements (e.g., pixels) to numerical values based on the underlying chart coordinate system, capturing the correct slope, offset, and unit. This mapping is highly sensitive to visual features such as tick density, line width, anti-aliasing effects, font styles, dual-axis configurations, and polar layouts. To enhance the model's ability to interpret such fine-grained visual signals, we fully fine-tune the last four layers of the vision encoder. These layers produce the visual features that are passed into the multimodal projector, and their norm scales and positional sensitivity determine how the language model interprets numerical structure.

The multimodal projector plays a critical role in aligning visual features with the language model's token space. Its output scale and variance directly impact numerical stability during generation. As we adapt both the tail of the vision encoder and the language model, the projector must be trained to maintain alignment in feature distributions, ensuring consistent unit interpretation across different font styles and axis ranges. Therefore, we also fully fine-tune the multimodal projector.

To ensure that the model outputs well-formatted tables, which preserve correct column order, row counts, grouping structure, and units, we adapt the language backbone using LoRA [21]. Since this is primarily a formatting and control task, applying lightweight adaptation via LoRA is efficient and effective.

6.4 Experiment Settings

We conduct the experiments using the proposed two-stage training framework to evaluate its effectiveness in improving MLLM performance on chart data extraction tasks. The experimental design includes three configurations:

- **Full Framework.** Fine-tuning the base model with both the CSPE and CTA stages.
- **Ablation 1.** Fine-tuning the base model with only the CTA stage (ablation of stage 1). This setting serves as a baseline to assess the contribution of the CSPE stage.
- **Ablation 2.** Fine-tuning the base model with only the CSPE stage (ablation of stage 2). This setting isolates the effect of the CTA stage on overall performance.

Parameter Settings. Experiments are conducted on a server equipped with 4 NVIDIA A100 80GB GPUs. For the LoRA adapters, we use a rank of 8 and an alpha of 16 to adapt the language backbone. Combined with the fully unfrozen vision layers and multimodal projector, this setup results in approximately 1.70% of the total model parameters being trainable. We use AdamW as the optimizer, with a weight decay of 0.1 and betas set to (0.9, 0.95) for both stages. The learning rate is set to 1e-5 for the vision encoder, 5e-5 for the multimodal projector, and 1.5e-4 for the LoRA parameters. We use a batch size of 4 with gradient accumulation steps set to 4. Each stage is trained for up to 10 epochs.

Evaluation. We evaluate all trained models on the benchmark introduced earlier and report performance using Adaptive MAPE as the primary metric.

6.5 Results

The performance of different training configurations on our benchmark is summarized in Table 4.

Compare to Existing MLLMs. After training with our full framework, the model achieves the lowest Adaptive MAPE of 4.87%, significantly outperforming the base model Qwen2.5-VL 7B (16.01%) as well as larger models such as GLM-4.5V (5.94%) and Gemini 2.5 Flash (6.72%). Our model also shows consistent improvements across all chart types, with more balanced performance between Cartesian and polar coordinate systems. Notably, the Adaptive MAPE on radar charts improves substantially from 25.47% to 3.21%, indicating a stronger understanding of polar encodings after fine-tuning. The Adaptive MAPE of scatter plots drops to below 1% (0.85%), which is expected since scatter plots are effectively the simplest form of Cartesian charts, and the CSPE stage includes scatter-like samples with a single point. The format success rate also improves significantly from 89.58% to 99.11%, indicating better control over structured outputs. Considering the lightweight language backbone of Qwen2.5-VL 7B, this result is impressive.

Table 6: Comparison of our model, existing chart-specific models, and advanced MLLMs on RNSS and RMS.

Model	RNSS	RMS
ChartInstruct	7.69	-
ChartAssistant	19.59	-
MatCha	74.79	19.52
TinyChart	41.75	41.18
UniChart	56.60	43.58
Gemini 2.5 Flash	62.22	69.69
GLM-4.5V	71.45	73.47
Ours	**75.92**	**78.61**

We also compare our model with several chart-specific models, including ChartInstruct [49], ChartAssistant [52], UniChart [47], TinyChart [78], and MatCha [37]. Because these models are not able to produce well-formed tables under our prompt design, we follow the prompting strategy used by Islam et al. [29] and evaluate their outputs using the RNSS [48] and RMS [36] metrics. The results are presented in Table 6. For ChartInstruct and ChartAssistant, which fail to generate valid table structures but still output partial numeric values, we report only their RNSS scores. Across both metrics, our model achieves the best performance among all chart-specific baselines.

Ablation Study. Ablation results confirm that both stages of our framework contribute meaningfully to overall performance. When the Coordinate System Perception Enhancement stage is removed, performance drops to 5.94%, suggesting that fine-tuning solely on chart-table pairs is insufficient for learning coordinate-based reasoning. A foundational understanding of coordinate geometry is necessary for accurate value extraction. Conversely, removing

the Chart-Table Alignment stage leads to a performance of 11.17%, indicating that learning to output structured tables is also critical.

These results validate the effectiveness of ExChart in enhancing the chart data extraction capabilities of MLLMs. By explicitly targeting coordinate perception and structured output generation, our approach enables models to achieve high numerical accuracy and robustness across diverse chart types. But it is also important to note that an Adaptive MAPE of 4.87% does not imply that the model is already reliable enough to function as a fully automatic data extraction tool. Rather, this result shows that a lightweight 7B model, when trained with our progressive framework, can surpass large state-of-the-art MLLMs by a clear margin. This improvement demonstrates the potential of the approach: if substantial gains can be achieved at the 7B scale, further scaling in model capacity and training data may bring the error rate closer to the threshold required for standalone, reliability-critical applications.

At the current stage, the model alone cannot serve as a dependable extractor without human oversight. Following the design principles of mixed-initiative systems, a practical path toward reliability is to trade a small amount of efficiency for user verification and lightweight correction. In this paradigm, the model provides an initial extraction, and the human ensures correctness through targeted interaction. We then conduct a user study to assess whether our trained model can effectively support such a workflow.

7 User Study

In this study, we try to answer: *Can MLLMs be integrated into an interactive workflow that enables reliable chart data extraction through user verification and correction?* Although our 7B MLLM outperforms top-tier commercial proprietary models, interactive support remains essential for reliable chart data extraction. Even the best-performing models still produce errors exceeding 4%, which may not satisfy the accuracy requirements of certain analytical tasks. Moreover, due to the black-box nature of MLLMs, users cannot fully trust extracted values without verification.

Mixed-initiative workflows allow humans to validate and correct model outputs, avoiding the fragility of fully automatic pipelines. To evaluate whether an MLLM-assisted workflow is practically useful, we conducted a controlled user study[1] focusing on both usability and efficiency. Our goal is to determine whether users, with adequate interface support, can efficiently validate model-extracted values at pixel precision and whether they perceive such a workflow as reliable and suitable for real-world use.

7.1 Prototype System

To support verification and correction, the system must project model-extracted numerical values back onto the chart image. This requires establishing a mapping from chart coordinate space to image pixel space, enabling each value to be visualized at its corresponding position. This is conceptually the inverse of what traditional extraction tools compute. Interactive tools such as WebPlotDigitizer first ask users to calibrate two points along the chart axes to establish the coordinate system. After calibration, users extract data by clicking on marks. We adopt the same calibration strategy.

[1]The study has been approved by State Key Lab of CAD&CG, Zhejiang University.

Our prototype supports both Cartesian and polar coordinates. Users perform a brief calibration step, after which the system overlays model-extracted values onto the chart as movable markers. Users can click and drag these markers to correct errors, achieving pixel-level accuracy comparable to traditional tools. To simplify projection, we require the model to output tables in standardized formats for each chart type. Figure 10 shows an example interface for basic line charts.

7.2 Procedure

We recruited 12 participants (A1—A12, male: 9, female: 3, age: 21—27) from a local university. All participants were familiar with common chart types such as bar, line, and pie charts. They were compensated according to our university's guidelines. We sampled 24 real-world charts from our benchmark to better reflect practical scenarios. Twelve charts contained six data marks, and twelve contained fifteen. The set covered basic bar, stacked bar, grouped bar, basic line, grouped line, and pie charts. Each session lasted approximately 45 minutes and included five phases:

- **Introduction (3 min)**. Participants were introduced to chart data extraction and the goal of obtaining accurate numerical values.
- **System demonstration (8 min)**. We demonstrated calibration, verification, and correction using six example charts representing all chart types included in the task.
- **Practice (5 min).** Participants familiarized themselves with the interface using sample charts.
- **Extraction tasks (≈20min)**. Each participant processed all 24 charts in randomized order. For each chart, participants validated every data point and corrected errors when necessary. They could only proceed after an experimenter confirmed pixel-level accuracy. We recorded the task completion time for each chart.
- **Questionnaire and interview (8 min)**. Participants completed a SUS [5] questionnaire and three custom Likert-scale items, followed by a brief semi-structured interview.

7.3 Measures and Analysis

We employed a mixed set of quantitative and qualitative measures to evaluate how well the MLLM-based workflow supports reliable chart data extraction.

Quantitative Measures. We collected task completion time for each chart and administered a post-study questionnaire consisting of the SUS and three five-point Likert items: **Q1**. I am satisfied with the extracted data after validation and correction. (This captures the perceived final quality of the workflow.) **Q2**. I perceive the model-extracted data as accurate. (This measures trust in raw model output before correction.) **Q3**. I would like to use this tool the next time I need to extract data from charts. (This evaluates acceptance and potential real-world adoption.)

Qualitative Feedback. We also conducted brief semi-structured interviews focusing on (1) the ease of the calibration process, (2) the clarity of visual grounding during verification and correction. For analysis, two researchers reviewed interview notes and grouped them into recurring themes through discussion.

Figure 10: Using the prototype system to validate and correct extracted chart data for a basic line chart. Users first calibrate the chart; then all extracted data points are overlaid for verification. Users can drag any data point to correct its value if necessary.

7.4 Results

Task Completion Time. On average, participants completed one chart with six data points in 31.35 seconds (SD = 8.39) and one chart with fifteen data points in 32.87 seconds (SD = 9.32). To contextualize efficiency, we compared these results with the task times reported in Figure 7 of ChartSense. For bar charts, including grouped and stacked variants, our system enabled users to verify 15 data points in 31.73 seconds on average (SD=8.93), which is even faster than the approximately 34.5 seconds required for nine-point basic bar charts using ChartSense. Similarly, for basic line and grouped line, users completed fifteen-point verification in 36.31 seconds on average (SD=9.74), substantially faster than the approximately 57.5 seconds reported by ChartSense for only nine-point line charts. These comparisons suggest that combining MLLM-assisted automation with interactive correction yields a verification efficiency that surpasses that of prior mixed-initiative approaches.

System Usability. Participants rated the system highly on usability [3], with an average SUS score of 94.79 (SD=3.14). The three Likert-scale questions yielded positive outcomes: *Q1 (Final result satisfaction):* mean=4.92, SD=0.28; *Q2 (Perceived model accuracy):* mean=3.67, SD=0.47; *Q3 (Future use preference):* mean=5.00, SD=0.00. The results show that participants were satisfied with the final corrected output (Q1), and all participants expressed willingness to use the system in the future (Q3). However, Q2 scores were noticeably lower, suggesting that although the workflow is effective, participants do not fully trust the model's raw outputs without verification. This confirms the need for interactive validation as part of model-assisted extraction workflows.

Qualitative Themes. Analysis of the interview notes revealed two recurring themes related to the calibration, verification, and correction processes.

Calibration is easy to perform but remains the primary time cost. Most participants described the calibration step as intuitive and quick, yet still the dominant contributor to total task time. Several participants proposed improvements, such as enabling automatic detection of calibration points (A4) or allowing users to reuse calibration parameters for similar charts to avoid repeated setup (A8).

Visual grounding supports verification well, but can be refined. Participants generally found the overlay of extracted data points clear and helpful for checking accuracy. However, they also suggested enhancements. For instance, A2 proposed additional visual cues, such as small vertical lines indicating alignment with bar tops, to aid precision. A2 and A9 both recommended interaction techniques, such as snapping markers to the nearest visual boundary to ease fine adjustments.

7.5 Summary

The user study shows that an MLLM-assisted interactive workflow can effectively support reliable chart data extraction. Participants were able to verify and correct model-extracted values efficiently and rated the overall workflow highly on usability. At the same time, feedback indicates clear directions for interface refinement, particularly in improving calibration efficiency and enhancing visual grounding during verification. Overall, these results demonstrate that MLLMs could also benefit from collaborating with humans to provide a promising approach to chart data extraction.

8 Discussion

In this section, we discuss the broader implications of our findings and potential extensions of our approach.

8.1 Extending to Composite and Layered Charts

While our current work focuses on charts with single mark types (e.g., bars, lines, or points), many real-world visualizations involve composite or layered encodings, such as bar charts with overlaid line charts, bubble charts that combine scatter and size encodings,

or dual-axis plots. These charts introduce additional complexity in both coordinate system interpretation and mark-level disentanglement. ExChart is naturally extensible to these more complex cases. Given a composite chart type, we can identify its underlying coordinate system(s) and decompose the chart into atomic mark encodings. With this decomposition, we can generate corresponding training data for the CSPE stage. Then, CTA can be applied to train the model to extract structured data from the full chart, regardless of mark complexity.

8.2 Implications for Human-AI Workflows

Our findings suggest integrating MLLMs into existing chart digitization tools, such as WebPlotDigitizer, to support a mixed-initiative workflow that combines automation with user verification. An MLLM can first generate an initial candidate data table directly from a chart image, which is then preloaded into the tool as editable values or plotted points. This removes the need for users to begin with an empty canvas and manually place control points. In this setting, the interactive interface shifts from primary extraction to validation and correction. Tools like WebPlotDigitizer already support axis calibration and value adjustment, allowing users to identify and fix errors in ambiguous or unreliable regions. This creates a lightweight validation loop that fits naturally into existing practice: the model performs large-scale, repetitive extraction, while human effort is focused on judgment-intensive steps. The workflow reduces effort without sacrificing accuracy and keeps users in control of the final data. It positions MLLMs as complementary components that turn manual digitization tools into effective human-AI workflows.

8.3 Implications for General CQA

Our findings also carry important implications for CQA. If a model cannot accurately extract underlying data values from a chart, its ability to answer quantitative questions may rely on superficial cues or spurious correlations rather than true chart understanding. This raises questions about the reliability of current CQA benchmarks, which often overlook data perception quality. To address this, we introduce a label-free data extraction benchmark that offers a more targeted and interpretable evaluation of chart comprehension. By isolating the data perception component, our benchmark enables clearer diagnosis of whether errors stem from flawed perception or faulty reasoning. Future CQA training and evaluation could benefit from decomposing the chart understanding ability into two aspects: (1) the perception ability responsible for interpreting the underlying data of the chart, and (2) the reasoning ability operating over known values. The former could be improved upon using the strategies proposed in this work, while the latter could leverage existing, high-performing reasoning LLMs.

8.4 Future Work

This study opens several promising directions for future research:

Enhancing Chart Diversity. While the current benchmark already supports systematic comparison of chart data extraction performance across MLLMs and helps surface key failure modes, expanding it with more real-world charts and greater chart diversity would enable a more complete evaluation. For instance, it remains unclear how MLLMs behave on real-world charts affected by stronger distortions, such as low-quality screenshots or misaligned scans. Broader coverage of real-world charts would help expose weaknesses that are not well captured in partially synthetic settings. As future work, we plan to extend the benchmark with a wider range of real-world charts and apply additional augmentation techniques to simulate more severe distortions. In parallel, training models on real-world charts with more varied visual styles may improve robustness and help them handle unconventional or noisy design choices more reliably.

Exploring the Impact of Table Shape. We observed that value accuracy may correlate with the position of values in the output sequence. As a table can be transformed to multiple shapes, this motivates further investigation into how different table shapes, such as varying numbers of rows and columns or the presence of merged cells, affect model performance. Understanding these structural influences could inform the design of more effective table representations and extraction strategies.

Incorporating Mechanistic Interpretability. As MLLM performance improves, interpretability becomes more important. Users will want to understand how reliable the values extracted by the model are. Future work could involve integrating uncertainty estimation to provide confidence scores for individual predictions. This would help users assess the reliability of the extracted values. Additionally, exposing intermediate signals, such as attention distributions or predicted error likelihoods, could support automatic or interactive inference-time interventions [9, 41], enabling users to adjust model outputs when uncertainty is high.

9 Conclusion

In this work, we investigate whether MLLMs can serve as reliable chart data extractors. We introduce a comprehensive benchmark that evaluates chart data extraction across diverse chart types and visual styles. Using this benchmark, we systematically assess advanced MLLMs and find that, although they can generate well-structured tables, they still struggle with numerical accuracy. To address this challenge, we propose a two-stage training framework that mirrors the human chart-reading process, enabling models to learn chart data extraction progressively. Our approach leads to substantial improvements, achieving state-of-the-art performance on our benchmark. A user study further validates the effectiveness of our method, demonstrating that, when paired with human verification, our model supports accurate and efficient chart data extraction in practical scenarios. Overall, this work provides a detailed examination of current MLLM limitations and presents an effective framework for improvement, laying the groundwork for future research on reliable chart data extraction.

Acknowledgments

The work was supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LD25F020003, NSFC (62421003, 62402428), and Ningbo Yongjiang Talent Programme (2023A-396-G). The authors gratefully acknowledge the support of Zhejiang University Education Foundation Qizhen Scholar Foundation.

References

[1] Mubashara Akhtar, Nikesh Subedi, Vivek Gupta, Sahar Tahmasebi, Oana Cocarascu, and Elena Simperl. 2024. ChartCheck: Explainable Fact-Checking over Real-World Chart Images. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, Bangkok, Thailand, 13921–13937. doi:10.18653/v1/2024.findings-acl.828

[2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. 2025. Qwen2.5-VL Technical Report. arXiv:2502.13923 [cs.CV]

[3] Aaron Bangor, Philip T. Kortum, and James T. Miller. 2008. An Empirical Evaluation of the System Usability Scale. *International Journal of Human–Computer Interaction* 24, 6 (2008), 574–594. doi:10.1080/10447310802205776

[4] Ekaba Bisong. 2019. Matplotlib and Seaborn. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Apress, Berkeley, CA, 151–165. doi:10.1007/978-1-4842-4470-8_12

[5] John Brooke. 2013. SUS: a retrospective. *J. Usability Studies* 8, 2 (Feb. 2013), 29–40.

[6] Chengliang Chai, Guoliang Li, Ju Fan, and Yuyu Luo. 2021. CrowdChart: Crowdsourced Data Extraction From Visualization Charts. *IEEE Transactions on Knowledge and Data Engineering* 33, 11 (2021), 3537–3549. doi:10.1109/TKDE.2020.2972543

[7] Jinyue Chen, Lingyu Kong, Haoran Wei, Chenglong Liu, Zheng Ge, Liang Zhao, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. 2024. OneChart: Purify the Chart Structural Extraction via One Auxiliary Token. In *Proceedings of the 32nd ACM International Conference on Multimedia* (Melbourne VIC, Australia) *(MM '24)*. Association for Computing Machinery, New York, NY, USA, 147–155. doi:10.1145/3664647.3681167

[8] Nan Chen, Yuge Zhang, Jiahang Xu, Kan Ren, and Yuqing Yang. 2025. VisEval: A Benchmark for Data Visualization in the Era of Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (2025), 1301–1311. doi:10.1109/TVCG.2024.3456320

[9] Shiqi Chen, Tongyao Zhu, Ruochen Zhou, Jinghan Zhang, Siyang Gao, Juan Carlos Niebles, Mor Geva, Junxian He, Jiajun Wu, and Manling Li. 2025. Why Is Spatial Reasoning Hard for VLMs? An Attention Mechanism Perspective on Focus Areas. arXiv:2503.01773 [cs.CL]

[10] Zixin Chen, Sicheng Song, KaShun Shum, Yanna Lin, Rui Sheng, Weiqi Wang, and Huamin Qu. 2025. Unmasking Deceptive Visuals: Benchmarking Multimodal Large Language Models on Misleading Chart Question Answering. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Suzhou, China, 13756–13789. doi:10.18653/v1/2025.emnlp-main.695

[11] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. InternVL: Scaling up Vision Foundation Models and Aligning for Generic Visual-Linguistic Tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 24185–24198.

[12] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities. arXiv:2507.06261 [cs.CL]

[13] Wenjing Dai, Meng Wang, Zhibin Niu, and Jiawan Zhang. 2018. Chart decoder: Generating textual and numeric information from chart images automatically. *Journal of Visual Languages & Computing* 48 (2018), 101–109.

[14] Amit Kumar Das, Mohammad Tarun, and Klaus Mueller. 2025. Charts-of-Thought: Enhancing LLM Visualization Literacy through Structured Data Extraction. *IEEE Transactions on Visualization and Computer Graphics* (2025), 1–11. doi:10.1109/TVCG.2025.3634813

[15] Kenny Davila, Srirangaraj Setlur, David Doermann, Bhargava Urala Kota, and Venu Govindaraju. 2020. Chart mining: A survey of methods for automated chart analysis. *IEEE transactions on pattern analysis and machine intelligence* 43, 11 (2020), 3799–3819.

[16] Dazhen Deng, Yihong Wu, Xinhuan Shu, Jiang Wu, Siwei Fu, Weiwei Cui, and Yingcai Wu. 2023. VisImages: A Fine-Grained Expert-Annotated Visualization Dataset. *IEEE Transactions on Visualization and Computer Graphics* 29, 7 (2023), 3298–3311. doi:10.1109/TVCG.2022.3155440

[17] Yucheng Han, Chi Zhang, Xin Chen, Xu Yang, Zhibin Wang, Gang Yu, Bin Fu, and Hanwang Zhang. 2023. ChartLlama: A Multimodal LLM for Chart Understanding and Generation. arXiv:2311.16483 [cs.CV]

[18] Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. 2025. GLM-4.5V and GLM-4.1V-Thinking: Towards Versatile Multimodal Reasoning with Scalable Reinforcement Learning. arXiv:2507.01006 [cs.CV]

[19] Enamul Hoque, Parsa Kavehzadeh, and Ahmed Masry. 2022. Chart Question Answering: State of the Art and Future Directions. *Computer Graphics Forum* 41, 3 (2022), 555–572. doi:10.1111/cgf.14573

[20] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, USA) *(CHI '99)*. Association for Computing Machinery, New York, NY, USA, 159–166. doi:10.1145/302979.303030

[21] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR* 1, 2 (2022), 3.

[22] Kevin Hu, Michiel A. Bakker, Stephen Li, Tim Kraska, and César Hidalgo. 2019. VizML: A Machine Learning Approach to Visualization Recommendation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3290605.3300358

[23] Kevin Hu, Snehalkumar 'Neil' S. Gaikwad, Madelon Hulsebos, Michiel A. Bakker, Emanuel Zgraggen, César Hidalgo, Tim Kraska, Guoliang Li, Arvind Satyanarayan, and Çağatay Demiralp. 2019. VizNet: Towards A Large-Scale Visualization Learning and Benchmarking Repository. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3290605.3300892

[24] Kung-Hsiang Huang, Hou Pong Chan, May Fung, Haoyi Qiu, Mingyang Zhou, Shafiq Joty, Shih-Fu Chang, and Heng Ji. 2025. From Pixels to Insights: A Survey on Automatic Chart Understanding in the Era of Large Foundation Models. *IEEE Transactions on Knowledge and Data Engineering* 37, 5 (2025), 2550–2568. doi:10.1109/TKDE.2024.3513320

[25] Kung-Hsiang Huang, Mingyang Zhou, Hou Pong Chan, Yi Fung, Zhenhailong Wang, Lingyu Zhang, Shih-Fu Chang, and Heng Ji. 2024. Do LVLMs Understand Charts? Analyzing and Correcting Factual Errors in Chart Captioning. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, Bangkok, Thailand, 730–749. doi:10.18653/v1/2024.findings-acl.41

[26] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. GPT-4o System Card. arXiv:2410.21276 [cs.CL]

[27] OpenAI Inc. 2025. Introducing GPT-4.1 in the API. https://openai.com/index/gpt-4-1/.

[28] Mohaiminul Islam and Shangzhu Jin. 2019. An Overview of Data Visualization. In *2019 International Conference on Information Science and Communications Technologies (ICISCT)*. 1–7. doi:10.1109/ICISCT47635.2019.9012031

[29] Mohammed Saidul Islam, Raian Rahman, Ahmed Masry, Md Tahmid Rahman Laskar, Mir Tafseer Nayeem, and Enamul Hoque. 2024. Are Large Vision Language Models up to the Challenge of Chart Comprehension and Reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. Association for Computational Linguistics, Miami, Florida, USA, 3334–3368. doi:10.18653/v1/2024.findings-emnlp.191

[30] Hyeon Jeon, Hyunwook Lee, Yun-Hsin Kuo, Taehyun Yang, Daniel Archambault, Sungahn Ko, Takanori Fujiwara, Kwan-Liu Ma, and Jinwook Seo. 2025. Unveiling High-dimensional Backstage: A Survey for Reliable Visual Analytics with Dimensionality Reduction. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 394, 24 pages. doi:10.1145/3706598.3713551

[31] Daekyoung Jung, Wonjae Kim, Hyunjoo Song, Jeong-in Hwang, Bongshin Lee, Bohyoung Kim, and Jinwook Seo. 2017. ChartSense: Interactive Data Extraction from Chart Images. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 6706–6717. doi:10.1145/3025453.3025957

[32] Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. 2018. DVQA: Understanding Data Visualizations via Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[33] Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Akos Kadar, Adam Trischler, and Yoshua Bengio. 2018. FigureQA: An Annotated Figure Dataset for Visual Reasoning. arXiv:1710.07300 [cs.CV]

[34] UW Interactive Data Lab. 2025. Vega-Lite Example Gallery. https://vega.github.io/vega-lite/examples/.

[35] Yijie Lian, Jianing Hao, Wei Zeng, and Qiong Luo. 2025. A survey of visual insight mining: Connecting data and insights via visualization. *Visual Informatics* 9, 4 (2025), 100271. doi:10.1016/j.visinf.2025.100271

[36] Fangyu Liu, Julian Martin Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. 2023. DePlot: One-shot visual language reasoning by plot-to-table translation. arXiv:2212.10505 [cs.CL]

[37] Fangyu Liu, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Yasemin Altun, Nigel Collier, and Julian Eisenschlos. 2023. MatCha: Enhancing Visual Language Pretraining with Math Reasoning and Chart Derendering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Toronto, Canada, 12756–12770. doi:10.18653/v1/2023.acl-long.714

[38] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved Baselines with Visual Instruction Tuning. In *Proceedings of the IEEE/CVF Conference on*

Computer Vision and Pattern Recognition (CVPR). 26296–26306.
[39] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. https://llava-vl.github.io/blog/2024-01-30-llava-next/
[40] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems* 36 (2023), 34892–34916.
[41] Sheng Liu, Haotian Ye, Lei Xing, and James Zou. 2024. Reducing Hallucinations in Vision-Language Models via Latent Space Steering. arXiv:2410.15778 [cs.CV]
[42] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. 2025. MMBench: Is Your Multi-modal Model an All-Around Player?. In *Computer Vision – ECCV 2024*. Springer Nature Switzerland, Cham, 216–233.
[43] Automeris LLC. 2024. WebPlotDigitizer. https://automeris.io.
[44] Junyu Luo, Zekun Li, Jinpeng Wang, and Chin-Yew Lin. 2021. ChartOCR: Data Extraction From Charts Images via a Deep Hybrid Framework. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 1917–1925.
[45] Yuyu Luo, Jiawei Tang, and Guoliang Li. 2021. nvBench: A Large-Scale Synthesized Dataset for Cross-Domain Natural Language to Visualization Task. arXiv:2112.12926 [cs.HC]
[46] Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. 2025. ChartQAPro: A More Diverse and Challenging Benchmark for Chart Question Answering. arXiv:2504.05506 [cs.CL]
[47] Ahmed Masry, Parsa Kavehzadeh, Xuan Long Do, Enamul Hoque, and Shafiq Joty. 2023. UniChart: A Universal Vision-language Pretrained Model for Chart Comprehension and Reasoning. arXiv:2305.14761 [cs.CL]
[48] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. 2022. ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*. Association for Computational Linguistics, Dublin, Ireland, 2263–2279. doi:10.18653/v1/2022.findings-acl.177
[49] Ahmed Masry, Mehrad Shahmohammadi, Md Rizwan Parvez, Enamul Hoque, and Shafiq Joty. 2024. ChartInstruct: Instruction Tuning for Chart Comprehension and Reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics, Bangkok, Thailand, 10387–10409. doi:10.18653/v1/2024.findings-acl.619
[50] Damien Masson, Sylvain Malacria, Daniel Vogel, Edward Lank, and Géry Casiez. 2023. ChartDetective: Easy and Accurate Interactive Data Extraction from Complex Vector Charts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 147, 17 pages. doi:10.1145/3544548.3581113
[51] Gonzalo Gabriel Méndez, Miguel A. Nacenta, and Sebastien Vandenheste. 2016. iVoLVER: Interactive Visual Language for Visualization Extraction and Reconstruction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 4073–4085. doi:10.1145/2858036.2858435
[52] Fanqing Meng, Wenqi Shao, Quanfeng Lu, Peng Gao, Kaipeng Zhang, Yu Qiao, and Ping Luo. 2024. ChartAssisstant: A Universal Chart Multimodal Language Model via Chart-to-Table Pre-training and Multitask Instruction Tuning. arXiv:2401.02384 [cs.CV]
[53] Nitesh Methani, Pritha Ganguly, Mitesh M. Khapra, and Pratyush Kumar. 2020. PlotQA: Reasoning over Scientific Plots. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.
[54] Kushin Mukherjee, Donghao Ren, Dominik Moritz, and Yannick Assogba. 2025. EncQA: Benchmarking Vision-Language Models on Visual Encodings for Charts. *IEEE Transactions on Visualization and Computer Graphics* (2025), 1–11. doi:10.1109/TVCG.2025.3634249
[55] Donald A. Norman. 1994. How might people interact with agents. *Commun. ACM* 37, 7 (July 1994), 68–71. doi:10.1145/176789.176796
[56] Jason Obeid and Enamul Hoque. 2020. Chart-to-Text: Generating Natural Language Descriptions for Charts by Adapting the Transformer Model. arXiv:2010.09142 [cs.CL]
[57] PlotDigitizer. 2025. PlotDigitizer. https://plotdigitizer.com.
[58] Raian Rahman, Rizvi Hasan, Abdullah Al Farhad, Md. Tahmid Rahman Laskar, Md. Hamjajul Ashmafee, and Abu Raihan Mostofa Kamal. 2023. ChartSumm: A Comprehensive Benchmark for Automatic Chart Summarization of Long and Short Summaries. *Proceedings of the Canadian Conference on Artificial Intelligence* (June 2023). doi:10.21428/594757db.0b1f96f6
[59] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 341–350. doi:10.1109/TVCG.2016.2599030
[60] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. 2011. ReVision: automated classification, analysis and redesign of chart images. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 393–402. doi:10.1145/2047196.2047247
[61] Danqing Shi, Yao Wang, Yunpeng Bai, Andreas Bulling, and Antti Oulasvirta. 2025. Chartist: Task-driven Eye Movement Control for Chart Reading. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1167, 14 pages. doi:10.1145/3706598.3713128
[62] Benny Tang, Angie Boggust, and Arvind Satyanarayan. 2023. VisText: A Benchmark for Semantically Rich Chart Captioning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Toronto, Canada, 7268–7298. doi:10.18653/v1/2023.acl-long.401
[63] Yuan Tian, Weiwei Cui, Dazhen Deng, Xinjing Yi, Yurun Yang, Haidong Zhang, and Yingcai Wu. 2025. ChartGPT: Leveraging LLMs to Generate Charts From Abstract Natural Language. *IEEE Transactions on Visualization and Computer Graphics* 31, 3 (2025), 1731–1745. doi:10.1109/TVCG.2024.3368621
[64] Yuan Tian, Dazhen Deng, Sen Yang, Huawei Zheng, Bowen Shi, Kai Xiong, Xinjing Yi, and Yingcai Wu. 2025. NoteFlow: Recommending Charts as Sight Glasses for Tracing Data Flow in Computational Notebooks. arXiv:2502.02326 [cs.HC]
[65] Yuan Tian, Chuhan Zhang, Xiaotong Wang, Sitong Pan, Weiwei Cui, Haidong Zhang, Dazhen Deng, and Yingcai Wu. 2025. ReSpark: Leveraging Previous Data Reports as References to Generate New Reports with LLMs. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25)*. Association for Computing Machinery, New York, NY, USA, Article 181, 18 pages. doi:10.1145/3746059.3747644
[66] B. Tummers. 2006. DataThief III. https://datathief.org.
[67] Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, et al. 2024. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *Advances in Neural Information Processing Systems* 37 (2024), 113569–113697.
[68] Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. 2023. Multimodal Large Language Models: A Survey. In *2023 IEEE International Conference on Big Data (BigData)*. 2247–2256. doi:10.1109/BigData59044.2023.10386743
[69] Yifan Wu, Lutao Yan, Leixian Shen, Yunhai Wang, Nan Tang, and Yuyu Luo. 2024. ChartInsights: Evaluating Multimodal Large Language Models for Low-Level Chart Question Answering. arXiv:2405.07001 [cs.CL]
[70] Renqiu Xia, Haoyang Peng, Hancheng Ye, Mingsheng Li, Xiangchao Yan, Peng Ye, Botian Shi, Yu Qiao, Junchi Yan, and Bo Zhang. 2024. StructChart: On the Schema, Metric, and Augmentation for Visual Chart Understanding. arXiv:2309.11268 [cs.CV]
[71] Zhengzhuo Xu, Sinan Du, Yiyan Qi, Chengjin Xu, Chun Yuan, and Jian Guo. 2024. ChartBench: A Benchmark for Complex Visual Reasoning in Charts. arXiv:2312.15915 [cs.CV]
[72] Zhengzhuo Xu, Bowen Qu, Yiyan Qi, Sinan Du, Chengjin Xu, Chun Yuan, and Jian Guo. 2025. ChartMoE: Mixture of Diversely Aligned Expert Connector for Chart Understanding. arXiv:2409.03277 [cs.AI]
[73] Cheng Yang, Chufan Shi, Yaxin Liu, Bo Shui, Junjie Wang, Mohan Jing, Linran Xu, Xinyu Zhu, Siheng Li, Yuxiang Zhang, et al. 2025. ChartMimic: Evaluating LMM's Cross-Modal Reasoning Capability via Chart-to-Code Generation. arXiv:2406.09961 [cs.SE]
[74] Yilin Ye, Jianing Hao, Yihan Hou, Zhan Wang, Shishi Xiao, Yuyu Luo, and Wei Zeng. 2024. Generative AI for visualization: State of the art and future directions. *Visual Informatics* 8, 2 (2024), 43–66. doi:10.1016/j.visinf.2024.04.003
[75] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2024. A survey on multimodal large language models. *National Science Review* 11, 12 (11 2024), nwae403. doi:10.1093/nsr/nwae403
[76] Xingchen Zeng, Haichuan Lin, Yilin Ye, and Wei Zeng. 2025. Advancing Multimodal Large Language Models in Chart Question Answering with Visualization-Referenced Instruction Tuning. *IEEE Transactions on Visualization and Computer Graphics* 31, 1 (2025), 525–535. doi:10.1109/TVCG.2024.3456159
[77] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-Language Models for Vision Tasks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 8 (2024), 5625–5644. doi:10.1109/TPAMI.2024.3369699
[78] Liang Zhang, Anwen Hu, Haiyang Xu, Ming Yan, Yichen Xu, Qin Jin, Ji Zhang, and Fei Huang. 2024. TinyChart: Efficient Chart Understanding with Visual Token Merging and Program-of-Thoughts Learning. arXiv:2404.16635 [cs.CV]
[79] Xuanle Zhao, Xianzhen Luo, Qi Shi, Chi Chen, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2025. ChartCoder: Advancing Multimodal Large Language Model for Chart-to-Code Generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vienna, Austria, 7333–7348. doi:10.18653/v1/2025.acl-long.363
[80] Yuhua Zhou, Xiyu Meng, Yanhong Wu, Tan Tang, Yongheng Wang, and Yingcai Wu. 2023. An intelligent approach to automatically discovering visual insights. *Journal of Visualization* 26, 3 (2023), 705–722.