INSTRUCTION-TUNING FOR CHART COMPREHENSION AND REASONING

MEHRAD SHAH MOHAMMADI

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF ARTS

GRADUATE PROGRAM IN
INFORMATION SYSTEMS AND TECHNOLOGY
YORK UNIIVERSITY
TORONTO, ONTARIO
NOVEMBER 2024

Abstract

Charts provide visual representations of data and are widely used for analyzing information, addressing queries, and conveying insights to others. Various chart-related downstream tasks have emerged recently, such as questionanswering and summarization. A common strategy to solve these tasks is to fine-tune various models originally trained on vision tasks language. However, such task-specific models are not capable of solving a wide range of chartrelated tasks, constraining their real-world applicability. To overcome these challenges, we introduce ChartInstruct: a novel chart-specific vision-language Instruction-following dataset comprising if instructions generated with distinct charts. We then present two distinct systems for instruction tuning on such datasets: (1) an end-to-end model (2) a pipeline model employing a two-step approach. Evaluation shows that our instruction-tuning approach supports a wide array of real-world chart comprehension and reasoning scenarios, thereby expanding the scope and applicability of our models to new kinds of tasks.

Acknowledgements

I would like to start by expressing my sincere thanks to my thesis advisor, **Prof. Enamul Hoque Prince**, for his continuous support from the very beginning of my masters degree. His helpful feedback and advice have greatly improved this thesis and made it possible to submit to a respected forum. I am truly grateful for his guidance in conducting research in the field of ML/NLP. It has been a great honor to work with him over the last two years.

I also want to thank **Prof. Xiaohui Yu** for being a committee member for my thesis.

A special thanks to **Prof. Shafiq Joty**, one of the co-authors of this work, for his valuable feedback and comments. I also appreciate the support and collaboration of **Ahmed Masry** and **Dr. Rizwan Parvez** in this project.

Lastly, I am deeply thankful to my parents and my brother for their constant support and encouragement.

Preface

Preface This thesis work was primarily carried out by **Mehrad Shah Mohammadi** under the supervision of **Prof. Enamul Hoque Prince** and has been accepted for the ACL 2024 conference [1]. My role in this project included participating in brainstorming sessions, helping define the problem, Designing Prompts and curating the instruction set, conducting post-processing and analysis, running experiments, analyzing the results, and contributing to the writing of the manuscript. My coauthors, **Enamul Hoque**, **Shafiq Joty**, and **Rizwan Parvez** provided valuable feedback and support in the writing process. Additionally, coauthor **Ahmed Masry** contributed to proposing the model architecture, conducting experiments, analyzing results, and writing the manuscript.

[1] Ahmed Masry*[1], **Mehrad Shahmohammadi***, Md Rizwan Parvez, Enamul Hoque, Shafiq Joty. ChartInstruct: Instruction Tuning for Chart Comprehension and Reasoning, Submitted for 2024 Conference on Association for Computational Linguistics **(ACL 2024)**.

[1] Equal contribution.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Information visualizations, including bar charts and line charts, are crucial in data analysis, providing essential insights for making informed decisions [40]. Despite their importance, identifying key patterns and responding to complex queries in these visualizations can be challenging. Recent research has introduced tasks to support chart analysis, such as chart question answering [73, 48], summarizing insights from charts [81, 99], reasoning over chart images, fact-checking [10], and automated visual data storytelling [101].

Initial efforts in this domain have involved adapting models originally developed for language and vision tasks [90, 73, 56, 50]. However, these models often fall short for chart-specific tasks because they do not fully account for the unique structures of charts, such as the relationships among elements like bars, legends, and axes. More recent approaches, including UniChart [75], Chart-T5 [123], and MatCha [66], are tailor-made for chart analysis, incorporating visual and mathematical reasoning. Yet, their application remains limited by a focus on a narrow range of tasks and sources, affecting their utility in diverse real-world scenarios.

A promising direction to address these limitations is through instruction tuning, a method shown to enhance language models' alignment with user intentions across a broad spectrum of tasks, even those not encountered during training. This approach, exemplified by models such as InstructGPT [86], FLAN-T5 [24], Alpaca [7], Vicuna [6], and LLaMA-chat [112], demonstrates significant improvements in model performance and user-task alignment. Similarly, in vision-language tasks, instruction tuning has been applied to fine-tune models with visual instructions, yielding better outcomes in aligning with user needs [59, 26]. Nonetheless, instruction tuning's potential for enhancing chart comprehension and reasoning has yet to be fully explored. While there are concurrent efforts in this area [67, 35], they often lack the breadth and depth of instruction-tuning tasks necessary for comprehensive real-world chart understanding.

1.2 Our Approach

In this project, we present Chart Instruction Tuning, a novel framework designed to establish a foundation for a general-purpose chart comprehension and reasoning assistant using Vision-Language Models (VLMs). To achieve this, we have curated a unique chart instruction-tuning dataset comprising real-world charts sourced from 157 online platforms, showcasing a broad spectrum of visual styles. This dataset leverages the capabilities of advanced Large Language Models (LLMs) such as GPT-3.5 [8], GPT-4 [85], and Gemini-Pro[109] to address the specific challenges posed by chart data, which is markedly different from general multi-modal datasets.

Given the unique nature of charts, a structured methodology is essential for enabling VLMs to utilize the instruction dataset effectively, thereby optimizing their performance in chart analysis tasks. In light of this, we propose two innovative VLM configurations. The first configuration enhances the LLaVA architecture [59] by integrating the UniChart [75] vision encoder, which is pre-trained specifically on chart images. This adjustment aims to refine our model's visual processing capabilities, making it more adept at interpreting chart data. For language modeling within this setup, we explore three different models: Llama2 (7B), a decoder-only model [112], and Flan-T5-XL (3B), an encoder-decoder model [24], and Gemma2[110] which is also a decoder-based model.

Our second configuration employs a two-step pipeline strategy: initially extracting the underlying data table from the chart image, followed by feeding this data into the LLM. This approach provides a versatile range of models, ensuring that our systems can be tailored to meet a variety of real-world needs and computational constraints.

Through these advancements, we aim to significantly enhance chart comprehension and reasoning capabilities, offering robust solutions to navigate the complexities of visual data analysis.

In our study, we conducted a thorough evaluation of our models across four key benchmarks: ChartQA [76], Chart2Text [82], OpenCQA [48], and ChartFC [10], to gauge their performance in various chart understanding and reasoning tasks. The outcomes of these evaluations underscore the state-of-the-art capabilities of our models, highlighting their proficiency in navigating the complexities of chart data interpretation.

Additionally, through human evaluations, we have gathered evidence that supports the effectiveness of our instruction-tuning methodology. This approach not only enhances the models' performance in traditional chart comprehension and reasoning tasks but also significantly increases their flexibility. Consequently, our models demonstrate a remarkable ability to adapt to a wide range of real-world scenarios, underscoring their potential to accommodate an extensive variety of new tasks. This adaptability is crucial for developing general-purpose tools that can meet the diverse needs of users across different domains, making our work a valuable contribution to the field of automatic chart comprehension and reasoning.

1.3 Contributions

We expect following contributions from this thesis:

- *(i)* We introduce a new instruction-following corpus with real-world charts and a wide range of tasks by utilizing LLMs.
- *(ii)* We introduce two distinct systems specifically tailored for chart understanding tasks.
- *(iii)* We conduct evaluations that demonstrate the performance of ChartInstruct across existing chart-related benchmark tasks while also expanding its applicability to new tasks.

We have made the materials related to this project available in the following GitHub repository: https://github.com/vis-nlp/ChartInstruct.

1.4 Outline

In this section, we outline the structure of the chapters within this Thesis. Chapter 2 starts by exploring previous research in chart analysis and reasoning, vision-language instruction-tuning, and relevant downstream tasks in chart reasoning domain. Chapter 3 elaborates on our approach, presenting the data we use to pre-train our models, and introduce the process we use to generate instruction-following data for the instruction-tuning stage. In chapter 4, we present our training strategies and the two system architectures we intend to use for our models. In Chapter 5, we show our experiments, benchmarks we intend to evaluate on, and ablation studies. Chapter 6 depicts our conclusion and future work possibilities.

2 Literature Review

In this chapter, we will have a brief look over the related works done for each of the definitions and steps that we are going to have toward the completion of the project. The mentioned steps are chart modeling, visual instruction-tuning, and chart downstream tasks.

2.1 Chart Architectures and Modeling

2.1.1 General Architectures

Pre-trained models are crucial for various vision and language tasks, as highlighted in [31]. The development of these models entails three main steps: text input processing, visual data encoding, and the integration of both textual and visual information.

The text encoding phase often uses BERT-based encoders [28, 63, 62, 71, 88, 108, 61, 8], processing text through transformer layers to produce a context-rich representation [115].

For visual encoding, early methods used Fast-RCNN [95] to identify object features, while more recent approaches encode the entire image using technologies like ResNet [36] or ViT [29, 62, 61, 124].

Finally, to combine textual and visual features, studies either develop a fusion encoder or use a dual encoder approach [22, 54, 107, 108, 43, 62, 88]. This step also incorporates cross-modal pre-training tasks such as image-text matching, contrastive learning, visual question answering, and image-text alignment

[21, 60, 43, 88, 22, 118, 61, 68, 124].

Alignment Methods As mentioned in the last subsection, there were various implementations of image-text fusions. However, Blip-2[61] mentions that there is a misalignment between the objectives and the architectures of the models implemented for this goal. Blip-2 introduced Q-former to fill this gap, as a bridge to connect the vision space to the text, with the objective of finding features that can help the decoder generate text from the bridging module to improve the system's performance. However, Q-former is a large component by itself. As a result, [68] uses a Linear layer as the alignment layer, which is cheap and fast, compared to its predecessor, and works well. Also [124] shows that although it may seem that Q-former should have better performance in theory because of more parameters and including attention layers[115], the linear layer serves as a better alignment layer than the Q-former.

2.1.2 Modeling

Chart comprehension methodologies can broadly be categorized into two distinct groups. The first group consists of approaches that adapt language or vision-language models directly for chart analysis tasks [76, 74, 56]. These models, although versatile, often face challenges in achieving optimal performance due to the absence of chart-specific pretraining. This limitation can hinder their ability to fully understand and interpret complex chart data.

On the other hand, the second category includes models that are meticulously designed and pretrained for chart-specific functionalities, such as question answering and summarization [75, 66, 123]. While these models demonstrate improved performance in their specialized tasks, their narrowly focused pretraining can restrict their utility across a wider array of chart types and real-world applications. This specialization, though beneficial for certain tasks, might not cater to the diverse needs encountered in practical scenarios.

Recent advancements have seen the application of Large Language Models (LLMs) like GPT-3, Llama [112], and GPT-4 in chart-related endeavors. These methods typically involve a two-step process: initially, specialized models such as UniChart [75] and Deplot [65] are employed to extract data values from charts. Subsequently, these extracted values serve as input for LLMs, facilitating tasks like question answering and summarization. Despite the innovative use of LLMs, the dependency on either proprietary models [16, 85] or publicly available models that lack chart-specific training could constrain their overall effectiveness and adaptability.

2.2 Visual Instruction-Tuning

Instruction tuning within Large Language Models (LLMs) has demonstrated significant advantages in aligning these models more closely with human intentions and improving their ability to generalize across a variety of tasks [24, 87, 117, 7, 6]. This approach has been successfully adapted to the vision-language domain, leading to notable advancements in model performance and user interaction [59, 124, 121, 61, 26]. Despite these achievements, the application of visual instruction tuning specifically within the chart analysis domain remains relatively rare.

A handful of studies have ventured into multimodal instruction tuning for charts [67, 35], leveraging CLIP [89] for vision encoding. However, CLIP, being optimized for natural images, may not be ideally suited for the nuanced requirements of chart analysis. Furthermore, these studies often resort to using synthetic charts or a limited range of real-world charts for training, and their scope of instruction-tuning tasks is somewhat restricted.

Our contribution stands out by introducing two novel systems that are purposefully engineered for the chart domain. These systems are trained on a comprehensive collection of chart images gathered from a wide spectrum of real-world sources, ensuring a rich diversity in visual styles and types. In addition, we expect that our approach encompasses an extensive range of chart-related applications, significantly broadening the scope of tasks that our models can effectively handle. This strategic focus not only enhances the models' performance in interpreting and reasoning over charts but also ensures their applicability and adaptability to real-world scenarios, marking a substantial step forward in the field of chart comprehension and reasoning.

2.3 Chart Downstream Tasks

The growing interest in chart-related tasks underscores a significant shift towards more sophisticated methods of understanding and generating insights from charts. This domain encompasses a range of tasks, each addressing different aspects of chart comprehension:

Chart Question Answering (CQA) Most existing research predominantly addresses questions requiring factual responses (e.g., "Which country has the highest GDP?") [19, 44, 52, 93]. Answering factual questions involves the system's ability to analyze both the question and the chart, followed by performing various arithmetic and logical operations (such as value comparison or identifying extremes). Many studies have explored understanding and analyzing such questions using a combination of heuristic methods and syntactic parsing techniques[80, 97, 106]. Additionally, various machine learning approaches have been used for table question answering and feature extraction from chart images [104, 44, 93]. For example, a significant amount of research has employed recurrent neural networks (RNN) with Long Short-Term Memory (LSTM) architecture to encode questions [104, 44, 45, 78, 93, 126]. However, recent findings indicate that transformer architectures consistently outperform RNNs, especially for lengthy input sequences. This has prompted a shift, with some researchers adopting transformer architectures for answering factual questions related to charts [64, 74, 76, 102]. [75] introduces an end-to-end encoder-decoder model that is able to provide direct answers given a specific token at the beginning of a user's query. Transformers were the first step leading to the popularization of Large Language Models(LLMs) like GPT-3[15]. With this new trend, researchers began using instruction-tuned Large Language models that are able to generate direct or short answers upon the user's request.[1, 68, 59, 124].

Open-Ended Explanatory Question Answering (OpenCQA) Unlike factual questions, open-ended questions are exploratory in nature (e.g.,Why the country with the highest GDP change in 2016?), and typically the expected answer is an explanatory text. Generating such explanatory answers had been a challenging task before the introduction of large language models. The available architecture and data at the time made it hard for researchers to generate explanatory text for their answers. Earliest related works to this problem are the ones that automatically summarize the most important insights from chart inputs as text[27, 83, 46, 104, 92]. For example, [83] adopted a transformer-based model to generate an explanatory text describing the chart. Future works utilize such data-to-text generation approaches by taking the query as additional input[75]. The appearance of large language models which are trained on huge collections of text made this impossible task suddenly possible. These experts in generating made it possible for researchers to come up with models that are able to come up with architectures and models that generate answers and their explanations based on the input chart and query like GPT-4[1, 68, 124]. LLaVA[68] is one of the first open-source models that has been introduced in the field of Large Multimodal Models(LMMs). Its success has lead researchers to apply this method to a real-life medicine task[59]. Although these models are incredible, they are for general use, and they are not specifically trained for chart interpretation and comprehension. GPT-4 is an incredible general agent for daily questions and tasks, but it is only able to talk about general things and overall trends, and it is not as good as the work has been done specifically on charts. As a result, researchers tried to come up with such Large Multimodal models for charts. ChartLlama[35] is the first module that has specifically been trained for charts using a similar end-to-end approach to LLaVA. Shortly before, Domino[116] was introduced, which is a dual system involving an LLM module and an end-to-end encoder-decoder model having a conversation, each having a role to fulfill to break down the reasoning step similar to the chain-of-thought prompting using atomic operations.[119]

Chart-to-text Current systems for summarizing charts into text typically rely on the chart's image[42] or metadata [34, 84, 46]. Prior to deep learning advancements, initial methods often employed a two-stage process, initially selecting content using statistical tools, followed by summary generation through preset templates[94, 125]. However, these template-based systems often lacked adaptability and struggled to accurately reflect complex data patterns. Recently, deep learning techniques have become prominent [34, 84, 125, 42, 122, 42, 25, 105, 72, 18, 46, 92], demonstrating improved results over template-based methods.

Of the five main publicly available Chart-To-Text summarization datasets, the Chart2Text dataset[84], comprising a smaller dataset compared to the others, is the earliest. Its relatively small size limits the effectiveness of data-driven approaches. The SciCAP dataset[42], designed for captioning charts, is not ideal for methods dependent solely on metadata. The AutoChart dataset[125] relies on predefined templates, resulting in limited variability in chart descriptions. More recently, the Chart-To-Text dataset [46], which includes chart images, metadata, and human-generated descriptions, has emerged as the second largest dataset for this task from Statista and Pew websites. The last work on this field was the ChartSumm dataset[92] which is almost twice the size of the Chart-To-Text dataset, offering a more substantial resource for this research area. UniChart[75] introduces an end-to-end module that tries to predict the underlying table representing the input chart and then is able to perform some tasks based on the user's query including chart summarization.

In line with this emerging trend, researchers have started employing instruction-tuned Large Language Models capable of producing concise comprehensive summaries describing the given visual image, as discussed in various studies including [1, 68, 59, 124] upon's user's request. Unlike Visual Chart Question Answering(VCQA), here, GPT-4 is able to provide a summary that describes the input image that might only lack the exact values of the characteristics.

Chart-to-Table: The objective here is to convert visual chart data into structured data tables [23, 75], a task that involves accurately extracting and representing data in a tabular format.

Automated Fact-Checking for Images Evidence-based fact-checking aims to predict claims veracity given evidence data. Considering that both claims and evidence can be communicated through various modalities, there has been a growing interest in Automated Fact-Checking (AFC) with images, as evidenced by recent studies [79, 17, 20, 120, 100]. Prior research in this area has predominantly focused on identifying manipulated or counterfeit images, rather than verifying claims based on evidence[14, 51, 100, 9]. While the detection of altered or fraudulent images can often be achieved using the image alone, the process of claim verification necessitates a combined understanding of both the claim and the corresponding evidence. For the mentioned non-chart-related works, the common thing was the lack of clarity and transparency during the checking procedure. QACheck introduces a multi-module system that breaks the inference into multiple question steps that show how the model operates the give a verdict.

ChartFC[10] is the first work researching Automated Fact-checking (AFC) in charts. They use an OCR-based method to extract information from the chart image, comparing it with the input claim using the introduced ChartBert model which is the current SOTA in this field. The dataset presents certain limitations, such as being restricted to only bar-chart data and being synthetically created from data tables sourced from Wikipedia. To mitigate these issues, Chartcheck[11] dataset was created. It is a novel yet smaller dataset compared to its predecessor.

With the appearance of the new trend of applying large language model solutions to problems, here, Chart Fact Checking is not an exception. OpenAI's LLMs such as ChatGPT[8] and GPT-4[1] are great reasoners. As a result, GPT-4 can be used to provide verdicts for user's input charts and claims.

In our work, we not only evaluate our models across these diverse downstream tasks but also introduce a new dimension to chart reasoning through an innovative instruction data generation process. This approach allows us to explore and develop models capable of handling a broader spectrum of chart-related tasks, further advancing the field of chart comprehension and reasoning. By tackling these varied tasks, our models will be able to demonstrate their versatility and effectiveness in interpreting complex chart data, contributing valuable insights and methodologies to the research community.

2.4 Challenges and Opportunities

2.4.1 Acquiring Chart Data and Chart-Text Instruction-following Data

Collecting real-world data chart generated by human always come in handy for deep learning models. This helps the models better learn and generalize on their intended tasks. Currently, most of such data are either without any textual description or involve template-based descriptions which is not helpful to the model satisfy users' needs. Human-authored queries and questions can pose challenges to the model due to language variations, informal language with or without typos, and richer semantics. Moreover, there are not many benchmarks involving samples specifically asking queries regarding the chart elements instead of their interpretations. For instance, PlotQA[78], doesn't support such questions. Another aspect to mention is that there are only a few frameworks supporting conversational experiences regarding vision-language models. This indicates a gap in instruction-following data especially for chart-related tasks. As mentioned LLaVA[68] has set the first step for conversational models in vision-language domains, and ChartLlama[35], MMC[67], ChartAssistant[77] for chart-related tasks which are fairly recent works.

2.4.2 Efficient Methods for Alignment

Although there are several ways of aligning image features to text such as Q-former[61], Linear Layer[68], gated cross-attention[12], There is still uncertainty around the fact whether these are the best ways for aligning text features with the visual features. These works show that although the complexity of the alignment method goes up, the performance might not improve. As a result, it is possible to believe that there might be a way to improve both the complexity of the method and the performance of the whole architecture.

2.4.3 Addressing chart data extraction challenges

The challenge in chart interpretation tasks often lies in the accurate extraction of chart data and visual encodings, essential for tasks requiring perceptual and arithmetic reasoning with charts. Visual encodings refer to how data is translated into visual elements of the chart. It's commonly assumed in many natural language interfaces that the data table and visual encodings of charts are easily accessible, as indicated by sources [41, 97, 98, 106]. However, this assumption is often not valid for many real-world charts found on the web. These charts are typically available in bitmap image format and lack accompanying underlying data.

3 Chart Instruction Data Generation

We build an instruction tuning dataset for enhancing VLMs' capabilities in tackling diverse understainding and generation tasks related to chart analysis. In this section, we describe the chart corpus collection followed by the instruction tuning data generation process. Figure 1 provides an overview of the instruction tuning process.

3.1 Chart Corpora Collection

Our objective is to create a comprehensive and varied chart dataset utilizing data from the real world to improve the generalizability of our model. To achieve this, we gather chart images from two primary sources: existing public datasets and charts obtained through web crawling. From the available datasets, we selected UniChart [75] for its extensive and varied pretraining corpus, which includes 611K charts along with metadata such as data tables, titles, and captions, offering a rich resource for our needs (for more information, see [75]). Nonetheless, the charts in UniChart predominantly originate from specific online sources like Pew [4], Statista [5], OECD [2], and OWID [3], which somewhat restricts the diversity in visual styles and data domains.

To overcome this diversity shortfall, we introduce a novel corpus named WebCharts, comprising **41K** chart images representing a wide variety of styles. This initiative began with a selection of web domains known to feature charts [39], followed by the aggregation of top image search results from these domains using

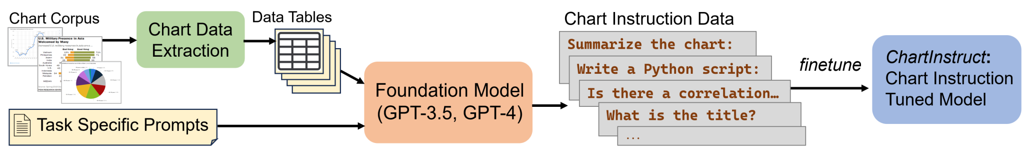


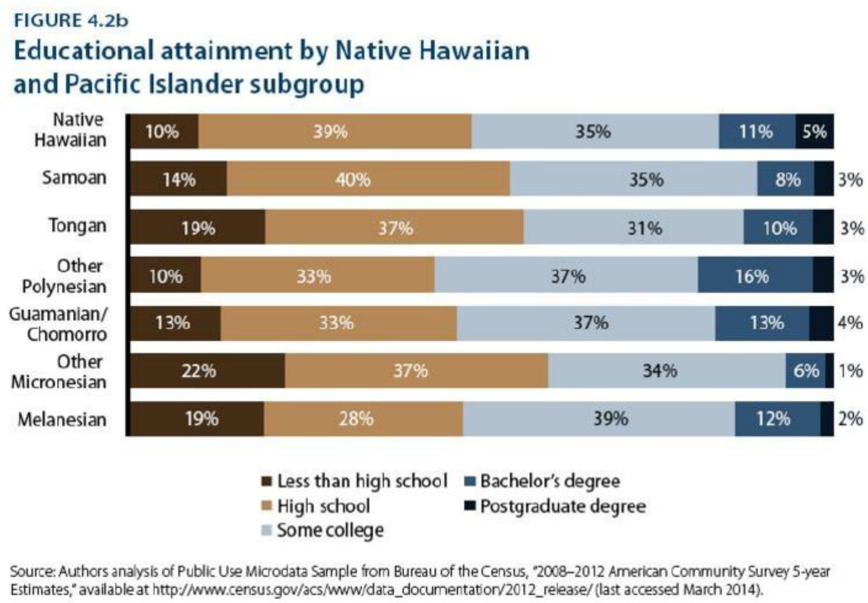
Figure 1: Instruction tuning process for chart collection. For the WebChart Corpus, the chart data is extracted automatically using Gemini Vision Pro. For distilling new tasks we use GPT-4, for other task generation we either use GPT 3.5 or GPT 4.

terms such as "chart images," "graphs," and "visual data". Subsequently, we implemented a binary VIT classifier [30] to sift chart images from non-chart content in our search findings. This step was complemented by a manual cleanse to eliminate any residual non-chart images, ensuring the purity of our dataset. A challenge we encountered with these web-crawled charts was the absence of associated data tables, which are indispensable for generating instructions across various chart data analysis tasks. To remedy this, we employed Gemini Pro Vision [109] for the automated extraction of data tables and chart titles. Our preference for Gemini was driven by its cost-effectiveness and the provision of an unlimited API rate, making it an ideal choice for our project. Figure 2 shows an example on Gemini[109] generating an underlying data table for an input image with the in JSON format given relevant prompt.

3.1.1 Instruction Data Generation

To improve the performance of Large Language Models (LLMs) in chart-related tasks through instruction tuning, we created a comprehensive chart instruction dataset. This dataset consists of 190,774 instructions for 70,882 charts, encompassing a wide range of chart comprehension and reasoning aspects, illustrated in

Input Image:



Gemini Output:

{ "data": [
["Group", "Less than high school", "High school", "Some college", "Bachelor's degree", "Postgraduate degree"],
["Native Hawaiian", 10, 39, 35, 11, 5],
["Samoan", 14, 40, 35, 8, 3],
["Tongan", 19, 37, 31, 10, 3],
["Other Polynesian", 10, 33, 37, 16, 3],
["Guamanian/Chomorro", 13, 33, 37, 13, 4],
["Other Micronesian", 22, 37, 34, 6, 1],
["Melanesian", 19, 28, 39, 12, 2
]}

Figure 2: shows an example where Gemini correctly comes up with an underlying data table representing the data provided in the input image.

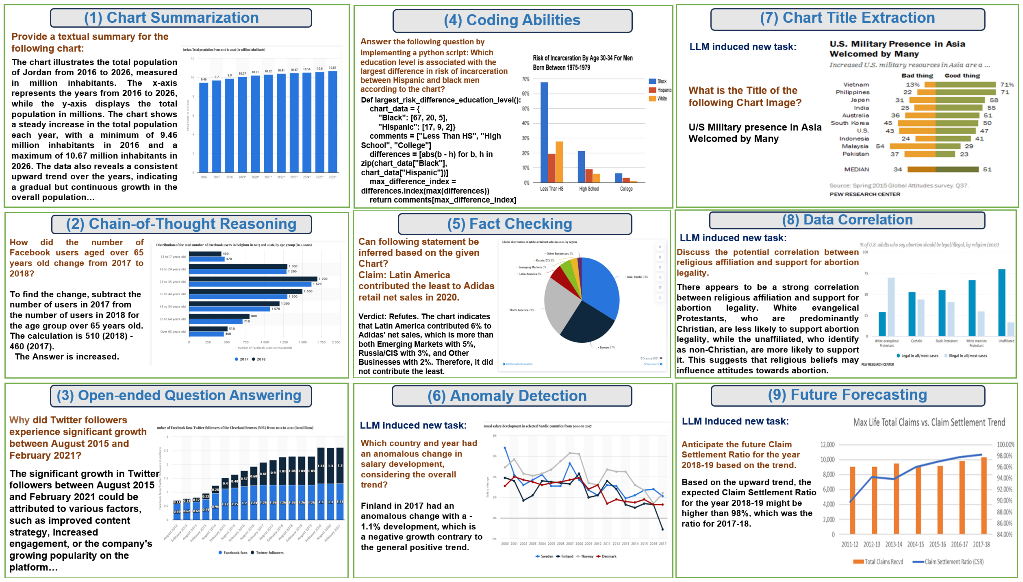


Figure 3: Examples of different chart-related tasks from our generated instruction dataset. Examples 1-5 are generated based on predefined tasks similar to previously developed downstream tasks like chart summarization, chart question answering, and fact-checking, while examples 6-9 introduce new types of tasks distilled by LLMs.

Figure 3. Here, we outline the methodology for generating this instruction data.

Task Selection for diverse reasoning and comprehension: our objective was to cover a broad spectrum of chart reasoning and comprehension tasks. We selected tasks aligned with traditional downstream tasks like chart summarization and question answering (QA), and introduced additional tasks such as code generation and Chain of Thought (CoT) reasoning. We also encouraged LLMs to suggest new tasks to further diversify the dataset. A brief overview of these tasks includes:

- The summarization task aims to generate a chart caption that captures the key insights such as trends and patterns from a given chart[50]. We also include Open-ended QA[49] in which the model generates an explanatory answer to the given question about a chart.

- Fact Checking: task [10] is included to improve our models ability to reduce errors and interpret chart data accurately. It takes a claim as input and then generates the verdict (refute or accept) along with an explanation.

- Chain-of-Thought (CoT) Reasoning: Focused on improving complex mathematical and visual reasoning. This includes variable dependent questions, inspired by ToolFormer [96] to remember and reuse previous calculations, and Variable Independent Questions for retrieval, comparison, and basic math analysis.

- Code Generation: Creating executable Python scripts to respond to user queries, influenced by the successes of PAL [33]. Here, the goal is to generate an independent function that receives no input, for our input query that returns the answer to the query by breaking down the problem to smaller parts handled by python commands.

- Novel Tasks: Introducing tasks requiring unique forms of reasoning and analysis, such as future value predictions, formulated by prompting an LLM to propose varied chart-related tasks.

Prompt Design for Instruction Creation We crafted a series of prompt templates for different tasks, each comprising a task description, the input chart data

Downstream Task/Model	GPT-3.5	GPT-4
Chart Summarization	✓	✗
Open-ended QA	✓	✗
Fact Checking	✓	✗
Chain-of-Thought Reasoning	✗	✓
Code Generation	✓	✓
Novel tasks	✗	✓

Table 1: Models used to generate instruction data for each different task. Choices are based on task complexity and costs.

table with metadata like the chart title, output constraints (if any), and the desired output format. An illustrative prompt example is depicted in Figure 5.

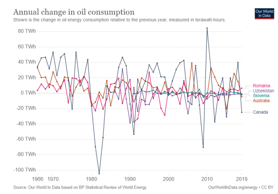
Generating inputs and outputs: Following prompt creation, we generated instructions by forming input-output instances for each template. We combined each chart's data table and title with a corresponding prompt, utilizing OpenAI's GPT3.5-Turbo and GPT4, as detailed in Table 1. The selection between these models depends on task complexity, with GPT-4 reserved for more intricate reasoning tasks and GPT3.5 for moderately complex ones. To economize on generation costs and enrich the variety of generated samples, we instructed LLMs to produce multiple samples for each chart in our prompts. Readers can find an example of multiple generated question/answer pairs in Figure 2. The prompts used for different tasks are also available in Appendix 6.2.

3.2 Dataset Analysis

We present key statistics and analyze the diversity and quality of the instruction dataset.

Summarization Prompt for GPT-3.5

Prompt:



Summarize a chart based on a provided data table, highlighting key aspect such as data background, maximum and minimum values, value comparisons, and trends across categories. The summary should not directly mention the input table, instead referring to the 'chart' or 'chart elements'. You'll also be given the title of the chart to get a context on what is shown in the chart. Do not use the title as it is like "the chart is titles TITLE", always try to infersomething new based on it. Include a brief explanation of the elements represented on the x-axis and y-axis. The entire summary should be cohesive, in paragraph form without bullet points, and not exceed 150 words. Conclude with a final sentence that encapsulates the most significant findings from the chart.
Title:
Title: Annual Change in Oil Consumption
Table:
Characteristic Romania Uzbekistan Slovenia Australia Canada
1966 5.31, -, -, 34.29, 35.92
1970 11.43, -, -, 28.99, 54.29
1980 -10.61, -, -, -13.87, -15.1
1990 19.19, -36.32, -, 7.35, -12.24
2000 7.35, 1.64, 0.01, -0.81, 1.64
2010 4.5, -5.71, 1.64, 34.29, 31.84
Extracted Instruction:
Summary:
The chart depicts the annual change in oil consumption across five countries over several decades. On the x-axis, time spans from 1966 to 2010, while the y-axis represents the percentage change in oil consumption. Each line on the chart represents a different country: Romania, Uzbekistan, Slovenia, Australia, and Canada. Maximum values are observed in Australia and Canada in various years, while Romania shows relatively stable fluctuations. Uzbekistan experiences significant fluctuations, including a notable decrease in 1990. Slovenia and Canada display a generally increasing trend. Overall, Australia and Canada exhibit the highest peaks and most consistent growth in oil consumption over the period analyzed.

Table 2: An example of prompt for Summarization Instruction Generation using GPT-3.5-turbo and an extracted sample

Statistics: Our chart corpus (WebCharts) is highly diverse, encompassing a variety of bar and line charts, pie and donut charts, and even unconventional chart types not prevalent in existing chart corpora (see Figure 7). The generated instructions set is dominated by chart summarization and open-ended QA to improve the chart comprehension ability but also augmented with reasoning tasks and creative new tasks generated by LLM (see Table 3). We placed particular emphasis on

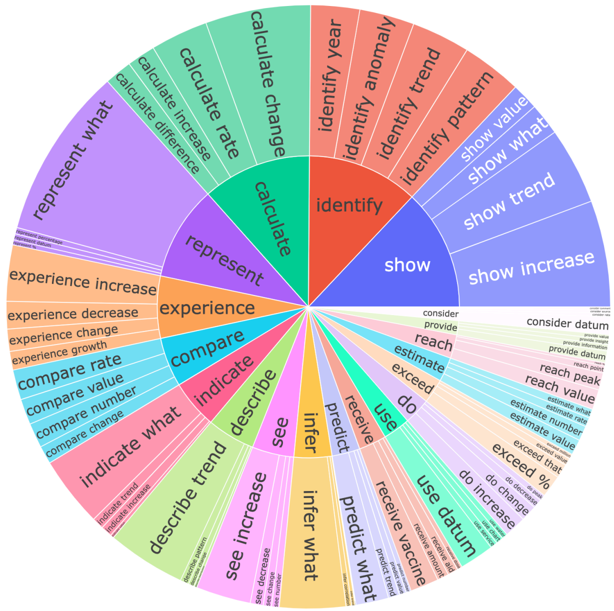


Figure 4: Top 20 most common root verb (inner circle) and corresponding four object verb pairs for all the generated instructions of our dataset.

WebCharts dataset due to its diversity, constituting 67.5% (157,190 samples) of our dataset.

Diversity: To investigate the diversity of gener- ated instructions, we employed the Berkeley Neural Parser[13] to identify the verb closest to the root along with its first direct noun object in each instruction. The analysis reveals a broad spectrum of comprehension and reasoning tasks expressed in the instructions (Figure 3). We further analyzed the newly proposed tasks generated by GPT-4 by clustering the instructions using the K-means algorithm. From Table 4, we observe that "Pattern and Outlier Detection" is the most common type of task, followed by various statistical analyses. Notably, the dataset includes interesting tasks not typically captured by existing downstream tasks, such as identifying correlations,

'Input'

'Task Description'

Analyze a chart and generate pairs of claims and verdicts about its data. Avoid using terms like 'rows', 'columns', or 'elements' from the data table; refer to 'chart' or 'chart image' instead. Each response must state whether it 'supports' or 'refutes' the question, followed by a brief explanation.

'Datatable and other metadata'

Title: Premier league table

Team	P	W	Pts
Liverpool	10	8	26
Man City	10	7	23
Chelsea	10	6	20
Man United	10	5	19
Arsenal	10	5	17

'Constraints':

Come up with 2-3 pairs of Claims and Verdicts with no additional text.

'Output Format'

Put ~ before starting each claim, and ^ before each verdict.

Output:

~ Liverpool has the highest number of points among the listed teams.
^ Supports: Liverpool indeed has the highest points (26) among the listed teams, indicating their strong performance relative to others.

~ Manchester City has the most wins among the listed teams.
^ Refutes: While Manchester City has a respectable number of wins (7), Liverpool actually has more wins (8), making them the team with the most wins in the chart.

~ Arsenal has fewer points than both Manchester United and Chelsea.
^ Supports: Arsenal has 17 points, which is fewer than Manchester United's 19 points and Chelsea's 20 points, indicating they are currently positioned below these teams in terms of pts.

Figure 5: An example prompt to LLM and the corresponding output for a fact-checking task. The input consists of the task description, chart data, any output constraints, and output format.

predicting values and trends, and distribution analysis. Overall, it suggests that the generated instruc tions set is indeed diverse and creative. We also visualize diversity in the length of the instructions inputs, and instance outputs in Figures 8 and 9.

Quality: We asked an expert annotator to evaluate the quality of the generated data on a random set of 100 instructions. We find that in general, the instruc-

Dataset	CoT Reasoning	Chart Summarization	Fact Checking	Open-ended QA	Coding Abilities	Novel Tasks	#Unique Charts
Statista	4,363	4,159	4,188	4,906	2,348	-	9992
PlotQA	4,159	3977	4,333	12,105	2,306	-	8199
OECD/OWID	4290	3999	4,080	13,213	2,994	-	10949
WebCrawled	14,459	41741	11,574	12,246	11,924	23,410	41742
Total	27,271(14.3%)	53,876(28.24%)	24,175(12.67%)	42,470(22.26%)	19,572(10.26%)	23,410(12.27%)	70,882

Table 3: The number of generated examples for each tasks based on data samples of the mentioned dataset. Some of the charts are used in multiple tasks. On the last column, we show the number of distinct charts used for instruction generation samples.

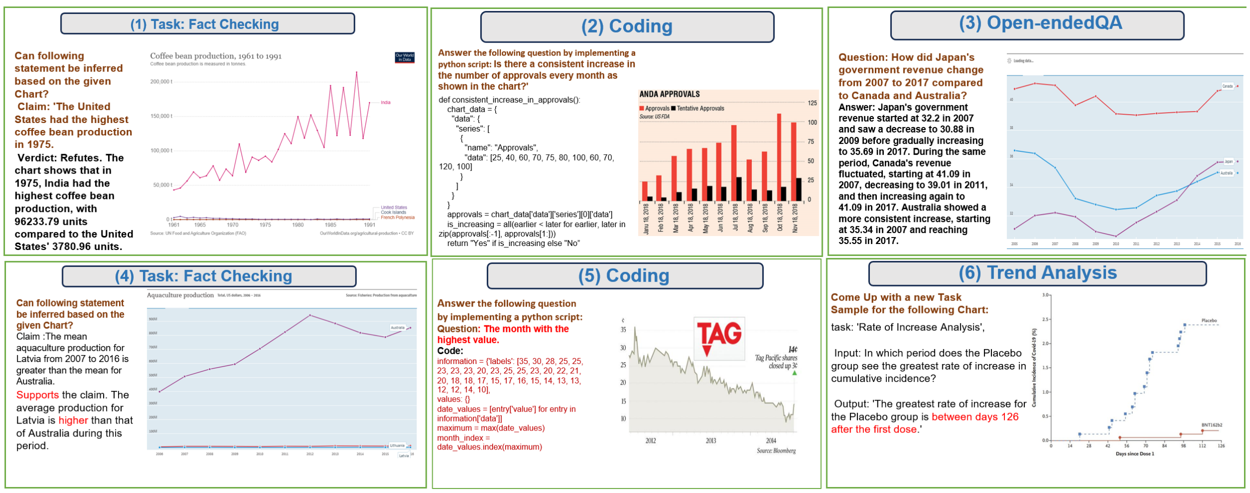


Figure 6: Few generated instruction samples. The Top row shows correct generated instructions. The bottom row shows some of the instructions involve errors: (4) generated wrong answer, (5) not following the description - wrong answer, (6) wrong answer. The errors are shown in red.

tions describe a valid task (87%) and the input matches the task description (86%) among generated instructions. In 61% and 8% cases, outputs for the generated inputs were fully and partially correct respectively. We list a number of correct and incorrect examples in Figure 6. We note that even when the outputs may be incorrect (e.g., contain factual errors), the corresponding task instructions can provide informative training signal as found by others (e.g., [38]).

Task Group	**#Examples**
Pattern and Outlier Detection	16,977
Statistical Analysis	9,148
Extremum Identification	6,545
Item Correlations	6,142
General Comparison	4,709
Relative Change Calculation	4,108
Time Series and Future Value Prediction	1,944
Data Point Identification	1,670
Performance and Result Analysis	1,396
Data Categorization	533
Distribution analysis	280

Table 4: The number of generated examples for various groups of new tasks created by GPT-4.

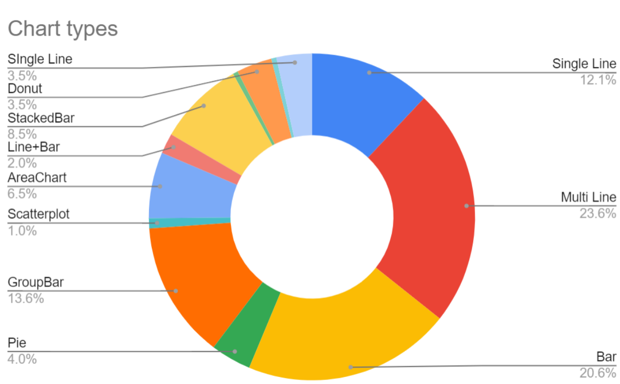


Figure 7: Chart types in WebChart Corpus.

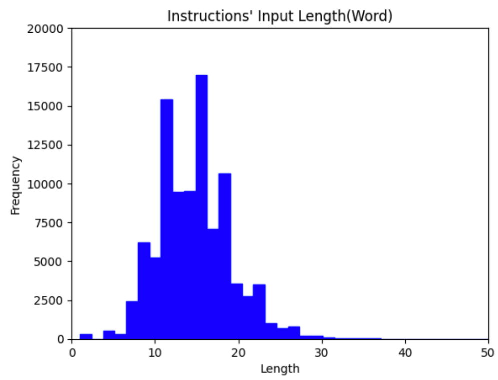


Figure 8: A histrogram of Instruction's Input Length distributions

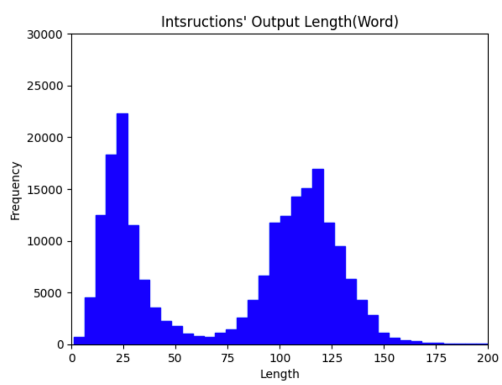


Figure 9: A histrogram of Instruction's Output Length distributions

4 Modeling, Training, and Evaluation Strategies

In this chapter, we will show our strategies for pre-training, instruction-tuning, and models that we are using. At first, we will provide a definition for our pre-training, instruction-tuning and fine-tuning problem. Next, we show the architectures we use, followed by the goals of each training stage. Afterwards, we introduce our downstream tasks and their means for human and automated evaluation methods that we used to evaluate the performance of our models on various tasks.

4.1 Problem Definition

In this section, we will first show how we model our different stages and tasks of our pipeline to a single format. Afterwards, we will explain how it will be applied to different tasks with various types of output.

Our goal is to prepare a pipeline that prepares models that receive a chart image, and a textual query explaining what the task is as input, to generate a textual answer label responding to the query based on the information available in the chart. As a result, we consider the different stages of our problem such as pre-training, instruction-tuning, and benchmark fine-tuning as a set of samples denotes as $S = \{C, Q, T\}$ in which C represents our input chart image, Q is a prompt or query regarding C, and L shows the expected output label following the prompt. You can see Figure 10 which shows an example on how we use the definition having a chart image and its query as inputs on the left, and the generated answer by our end-to-end model label on the right. Although there are more metadata that can be extracted from the charts such as title, description, etc. to be used for training, we use this mechanism for our models allowing them to function more aligned with real-world scenarios where most of the time chart image is the only information given to us about the chart. To close this up, we can see that based on our data generation pipeline, we can use the defined format.

4.2 Model Architectures

In this section, we first present the different modules we used use in our models and provide a brief description about them. Next, we introduce two variation of our Chart Instruction-tuned models. Based on our experiments, we were able to find two valid architectures following a slightly different route to generate the final output text, the usual end-to-end architecture having a **single-step** generation process, and our pipeline method involving **two-steps** of text generation in the process.

4.2.1 Employed Modules

- **UniChart**[75]: is a pre-trained model for chart comprehension reasoning. It is comprised of an image encoder and a text decoder. Having a similar problem structure as document understanding, to encode the input image, UniChart encoder builds upon donut[53], a state-of-the art document understanding model utilizing Swin Transformer[70] extended by multi-headed self-attentions and MLP layers to produce visual tokens. For the text-decoder, it utilizes BART[58] which is conditioned by the prompted text. For the encoder module, The input and outputs are an image, and embeddings resulted from visual features embeddings. For the decoder module, the input is the generated embeddings, and the output is text.

- **Flan-T5**[111] is an encoder-decoder text-to-text transformer model. Encoder-decoder models first processes input then starts generating. Flan-T5[111] improves upon T5[91] by being fine-tuned on task explicit instructions which help the model to better learn and follow new patterns of inputs and instructions.

- **LLaMA2**[114] is improving upon LLaMA[113] by being trained on a larger corpus comprised of 2 trillion token becoming a more optimized and efficient expansion on original transformer models[115]. It became a prominent open-sourced large language model that could show great performances same as ChatGPT[8] and GPT-4[1] considering that it is an open source model.

- **Gemma2**[110] is a variation of original Transformer model[115], receiving a variety of new attention mechanisms, activation function, and embedding model. Gemma2's smaller variations are trained using knowledge distillation from its larger models instead of next-token prediction.

4.2.2 ChartInstruct Architectures

End-to-end System: Our end-to-end system follows LLaVA [68] architecture. LLaVA incorporates CLIP [89] for visual encoding, an LLM for language generation, and an adapter module for transforming the encoded visual features to the LLM's input embedding space (Figure 10). LLaVA was originally designed for natural image understanding. We made following modifications to adapt it for chart understanding. First, we want substitute the CLIP vision encoder with the UniChart vision encoder [75], which is pretrained and optimized for chart image understanding. For the LLM, we are going to investigate two model types: the decoder-only architectures of LLaMA2 and Gemma2 [35] and encoder-decoder structure of Flan-T5 [24]. In the decoder-only decoder setups, projected visual features are injected directly into the language decoder, whereas in the Flan-T5 model, these features, along with the instructions, are first processed by the language encoder before the decoder generates a text. For the alignment module,

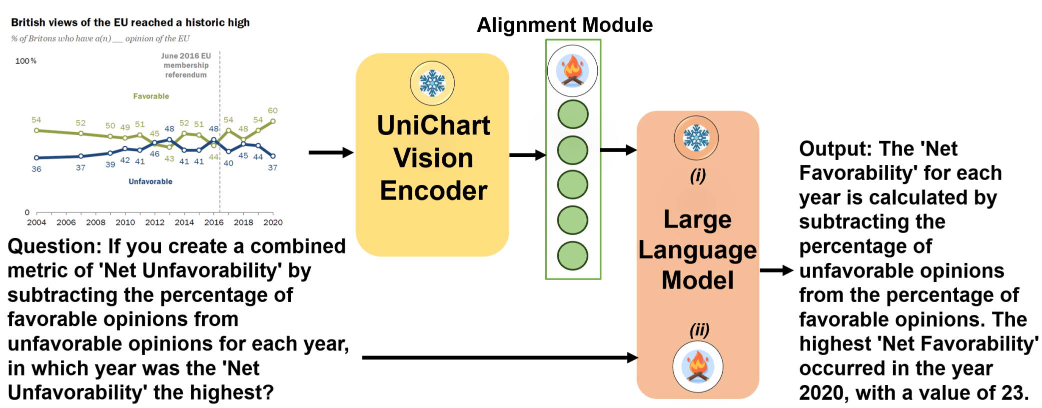


Figure 10: The architecture for our end-to-end system models: the LLM is frozen in our (i) pre-training step, while it updates its parameters in the (ii) instruction-tuning step. We either use Flan-T5-XL or Llama2 as LLM for this architecture.

we used a 3-layered multilayer perceptron to learn how to project visual features into decoder's embedding space. We experimented with the 7B variant of Llama2, 2B variant of Gemma2[110], and the 3B variant of Flan-T5 to provide a range of model sizes suitable for different applications. This architecture involves one text generation step, performed by the LLM at the end of the process.

Pipeline System: In this approach, we consider break down our problem by turning it into a two-step process. For our first step, we consider having a chart-to-table problem, which is then followed by a table-to-text process. As a result, a data table generation module will be first converting the chart image into a textual data table representation. Afterwards, The generated data table is going to be combined with the input instruction and fed into an LLM. We utilize the whole UniChart [75] model, which has been shown to be able to generate high-quality data tables from chart images, ensuring the textual representation closely mirrors the original chart's information. For LLM, we are going to conduct experiments with Llama2, Gemma2, and Flan-T5 models similar to our end-to-end approach. This architecture involves two text generation step. The first one is carried out by UniChart's text-decoder, and the second one is performed by the LLM at the end of the process.

4.3 Training Goals and Stages

In this section, we explain what we expect from each training stage for our models, and how we utilized our data to train our models. We use the weights provided

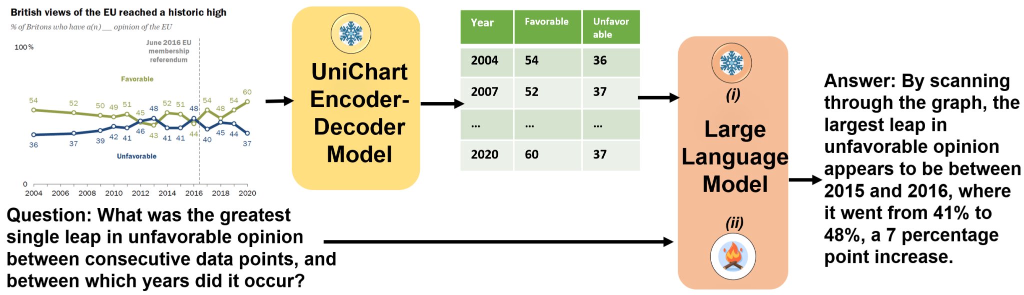


Figure 11: The architecture for our pipeline models: the encoder is frozen, while LLM updates its parameters.

for each model. As a result, our encoders and decoders are already experts in their own domain such as encoding charts and generating text. You can find the details on each stage's hyperparameters in Table 5

4.3.1 Pre-training stage(end-to-end system only)

To use a vision encoder and a text decoder for visual text generation, we need to align the resulting visual tokens from vision encoder with the embedding space used by the text decoder so that the decoder can comprehend the information. In order to do that, we use an MLP as the alignment module that has to learn how to this task. For that, first, we freeze the vision encoder and text decoder's parameters. Next, to expose the alignment module to two important skills needed for chart comprehension and reasoning, we use UniChart[75] dataset's chart summarization and data table generation tasks. Unlike the end-to-end system, this setup skips the alignment step since the visual features are not directly fed into the LLM(see figure11). Hence, we directly finetune the models on the instruction data.

4.3.2 Instruction-Tuning Stage

To make our model, specifically the language model, to become more versatile and better learn how to follow new input instructions, we train our model on the visual-instruction data generated in the last section. To do that, we also need to unfreeze the language model parameters so that it gets exposed to the new variations, keeping only the vision encoder frozen in the process. Other than the benchmark tasks such as chart question answering, open-ended question answering, and chart summarization, we wanted to improve the range of reasoning skills of these models. Hence, we considered new tasks for our instruction data such as chart fact checking, and LLM generated tasks including Data Correlation, Future Forecasting(see table4 for task types). To further improve our models, since language models are next-token prediction models, they are not best suited for responding to queries that require complex mathematical operations and reasoning. As a result, we also considered the code generation task inspired by PAL[32] and Chain-of-Thought reasoning[119] utilizing calculation tools[96].

4.3.3 Fine-Tuning Stage

To evaluate the models on different benchmark, we further fine-tune them on the training set of each benchmark respectively.

5 Experiments and Results

We evaluate the usefulness of **ChartInstruct** and demonstrate that our models built upon **ChartInstruct** achieve excellence in chart understanding and generation tasks. In addition to the existing downstream tasks, they also posit superior capabilities in new tasks. Below, we first discuss the setups, then experiments on downstream benchmarks and new chart tasks, and finally present an error analysis and challenges. Our evaluation complements the trivial quantitative approach based on automated metrics with detailed human evaluation on both seen and new tasks in multiple aspects, reflecting the true understanding of effectiveness of ChartInstruct . To ensure the reproducability of our research, we present the hyperparameters of our instruction tuning and downstream tasks experiments in Table 5. All experiments were conduced on a 4-A100 GPUs (80GB) machine.

5.1 Experimental Setup

5.1.1 Benchmarks

To assess the generalizability of our models across a spectrum of practical chart applications, we evaluate them on four established benchmark downstream tasks in the literature:

- ChartQA a factoid chart question answering dataset. We undertook the fine-tuning of our models using ChartQA[76], a benchmark designed to assess visual and logical reasoning abilities through questions. This task

	End-to-End System				Pipeline System			
Experiment	**# Epochs**	**Learning Rate**	**Batch Size**	**Hours**	**# Epochs**	**Learning Rate**	**Batch Size**	**Hours**
	Alignment							
Flan-T5-XL	4	2e-3	128	7	-	-	-	-
Llama 2	3	2e-3	64	24	-	-	-	-
Gemma 2	3	2e-3	128	6	-	-	-	-
	Instruction Tuning							
Flan-T5-XL	3	2e-5	32	8	3	1e-4	64	17
Llama 2	3	2e-5	32	20	3	1e-4	64	21
Gemma 2	3	2e-5	32	8	3	1e-4	64	15
	Finetuning on downstream tasks (Flan-T5-XL)							
ChartQA	10	1e-4	128	3	10	1e-4	128	7
OpenCQA	10	1e-4	128	1.5	10	1e-4	128	3
Chart-to-text Pew	10	1e-4	128	2	10	1e-4	128	3
Chart-to-text Statista	10	1e-4	128	4	10	1e-4	128	8
ChartFC	10	1e-4	128	1	10	1e-4	128	2
	Finetuning on downstream tasks (Llama 2)							
ChartQA	10	2e-5	32	6	10	1e-4	64	8
OpenCQA	10	2e-5	32	2	10	1e-4	64	4
Chart-to-text Pew	10	2e-5	32	2	10	1e-4	64	4
Chart-to-text Statista	10	2e-5	32	6	10	1e-4	64	9
ChartFC	10	2e-5	32	3	10	1e-4	64	3
	Finetuning on downstream tasks (Gemma 2)							
ChartQA	10	2e-5	32	2.5	10	1e-4	64	6
OpenCQA	10	2e-5	32	1	10	1e-4	64	2
Chart-to-text Pew	10	2e-5	32	1.5	10	1e-4	64	2
Chart-to-text Statista	10	2e-5	32	3.5	10	1e-4	64	7
ChartFC	10	2e-5	32	1	10	1e-4	64	2

Table 5: Training details for our instruction tuning and downstream tasks finetuning experiments.

specifically evaluates the model's proficiency in data retrieval and logical reasoning concerning numerical and visual inquiries.

- OpenCQA an open-ended chart question answering dataset. We further evaluated our pre-trained model by fine-tuning it on OpenCQA[47], a challenging benchmark in which the answers require explanatory answers. This task measures how comprehensive and related generated answers are given a chart and an open-ended question.

- Chart2Text a chart captioning dataset collected from two sources: Statista and Pew Research Center. We explored the abilities of our models to provide a concise text summarizing the information available in an image, by fine-tuning it on Chart-to-Text, a comprehensive benchmark on chart summarization involving a set of diverse and challenging chart images.

- ChartFC a chart fact-checking dataset. We conducted the final downstream task, evaluating our models on their fact checking abilities regarding an input image. We fine-tune our models on Fact-Checking, the first benchmark available on automatic fact checking for chart data.

5.1.2 Baselines

We compare ChartInstruct against seven baselines:

- **T5**[91], a unified seq2seq Transformer model that achieved state-of-the-art (SoTA) results on various text-to-text tasks, including question answering and summarization.

- **VL-T5**[22], a T5-based model that unifies Vision-Language (VL) tasks as text generation conditioned on multimodal inputs and achieved SoTA results on OpenCQA[49].

- **VisionTapas**[76], an extension of TaPas[37] for chart question answering.

- **ChartBERT**[11], a BERT-based model utilizing textual and visual information of charts for fact verification.

- **Pix2Struct**[57], a pretrained image-to-text model for visual language understanding, which can be finetuned on tasks involving visually-situated language, and achieved SoTA results on document understanding tasks.

- **MatCha**[66], an adapted version of Pix2Struct for charts that is further pretrained on math reasoning and chart data extraction tasks, achieving SoTA results on Chart-to-Text[99] and ChartQA[73].

- **UniChart**[75] achieving SoTA on Chart-to-Text, ChartQA, and OpenCQA.

5.1.3 Evaluation Metrics

We use Relaxed Accuracy (RA) for ChartQA, Accuracy for ChartFC, and BLEU for text-generation tasks (Chart-to-Text and OpenCQA). However, BLEU focuses mainly on n-gram matching, overlooking factors like informativeness and factual correctness. To address this, we conduct human evaluations to compare these aspects (see Section 5.3).

To evaluate the results on the ChartQA dataset, a relaxed accuracy measure is adopted for numeric answers, as proposed in previous work. This measure allows for a margin of error, considering that exact matches for numeric answers can be challenging due to potential inaccuracies in the data extraction process. Specifically, a numeric answer is deemed correct if it falls within 5% of the correct answer. However, for non-numeric answers, an exact match is required for them to be considered correct.

To evaluate our models on ChartFC, we consider the Accuracy metric. With it, we check whether our models correctly infer the needed information to generate a verdict whether the input fact is supported or refuted by the input chart.

To evaluate Open-CQA and Chart-to-Text, We use BLEU score. It quantifies the similarity between the machine-generated translations and the human references by examining the correspondence of words or sequences of words (n-grams) between the translations.

The BLEU score is defined as:

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{1}$$

where:

- p_n is the precision of n-grams,
- w_n is the weight for each n-gram (typically $w_n = \frac{1}{N}$ for uniform weights),
- BP is the brevity penalty, defined as:

$$BP = \{\, 1 \; if c > r, \; \exp\left(1 - \frac{r}{c}\right) if c \leq r, \tag{2}$$

- c is the length of the candidate translation,
- r is the effective reference length.

It measures the similarity the generated text is to one or the human references by comparing n-grams (sequences of words). BLEU calculates n-gram precision (Here four-grams) and includes a brevity penalty to avoid favoring overly short outputs. It results in a number between 0 and 1(also can be shown in percentage between 0 and 100) in which results closer to 1 mean perfect matching to the references.

5.2 Results and Findings

5.2.1 Main Result

We present the experimental results on downstream tasks in Table 6 compare with existing baselines. ChartInstruct models (both end-to-end and pipeline) outperforms previous state-of-the-art models, UniChart on all ChartQA and Chart-toText datasets. In particular, the Flan-T5-XL version excels on the ChartQA including the challenging human-written question set[73], which suggests that the model learned more complex mathematical and visual reasoning through the relevant instruction tuning tasks such as CoT reasoning, and coding abilities. ChartInstruct also achieved a higher BLUE score compared to UniChart on OpenCQA benchmark, which demonstrates our models capability to generate explanatory answers for questions about charts. Finally, ChartInstruct surpasses ChartBERT by a wide margin (8.85%) on the recently released fact-checking task. Overall, these results establish ChartInstruct as the SoTA model for chart comprehension and reasoning tasks.

Model	#Params	ChartQA (*RA*) aug.	human	avg.	OpenCQA (*BLEU*) OpenCQA	Chart-to-Text (*BLEU*) Pew	Statista	ChartFC (*Accuracy*) ChartFC
Open Source								
VisionTaPas [73]	-	61.44	29.60	45.52	–	-	-	-
T5 [73]	222M	56.96	25.12	41.04	9.28	10.49	35.29	-
VL-T5 [73]	-	56.88	26.24	41.56	14.73	-	-	-
ChartBERT [10]	-	-	-	-	-	-	-	63.8
Pix2Struct [56]	282M	81.6	30.5	56.0	-	10.3	38.0	-
Matcha[66]	282M	90.2	38.2	64.2	-	12.2	39.4	-
UniChart [75]	201M	88.56	43.92	66.24	14.88	12.48	38.21	-
Closed LMMs								
Gemini Pro [109]	-	-	-	74.1	6.84	28.5	25.8	65.8
GPT4-V [85]	-	-	-	78.5	3.31	35.9	18.2	69.6
End-to-End System								
ChartInstruct-Flan-T5-XL	3B	85.04	43.36	64.20	**16.71**	12.92	42.42	70.27
ChartInstruct-Llama2	7B	87.76	45.52	66.64	15.59	**13.83**	**43.53**	69.57
ChartInstruct-Gemma2	2B	85.12	44.84	64.98	15.10	12.68	43.14	67.79
Pipeline System								
ChartInstruct-Flan-T5-XL	3B	**93.84**	**50.16**	**72.00**	14.81	9.93	40.08	**72.65**
ChartInstruct-Llama2	7B	82.40	40.64	61.52	14.78	12.81	39.39	64.99
ChartInstruct-Gemma2	2B	82.80	41.44	62.12	14.75	10.47	38.90	64.39

Table 6: Evaluation results on four public benchmarks: ChartQA, Chart-to-Text, OpenCQA, and ChartFC. All the results are calculated after finetuning ChartInstruct.

Our observations reveal that the end-to-end system for ChartInstruct-LLama generally surpasses the corresponding LLaMA pipeline system across all benchmarks. This performance disparity is likely due to the fact that the data table alone does not capture all the nuanced information present in the charts, thus becoming a limiting factor in the pipeline systems effectiveness. Similarly, the end-to-end system of ChartInstruct-Flan-T5-XL performs better than the pipeline system on both OpenCQA and Chart-to-Text benchmarks. One notable exception is the reasoning-intensive tasks like ChartFC and ChartQA on which the pipeline FlanT5-XL system exhibits better performance. Furthermore, we notice that both

ChartInstruct-Flan-T5- XL and ChartInstruct-Llama achieve comparable performance, even tho the former has 4B fewer parameters. This efficiency makes ChartInstruct-FlanT5-XL more suitable for real-world applications with computational constraints.

To further assess the impact of our different instruction tuning tasks on our models performance, we conducted ablation studies on the ChartQA dataset using our best performing model, ChartInstruct-Flan-T5-XL (Pipeline System). Our ablation studies reveal that excluding tasks like Chart Summarization or Open-ended Question Answering results in a minor decline in performance.

You can find some of the examples properly generated by ChartInstruct-LLaMA in Figure 12.

5.2.2 Ablation studies

To understand the impact of the different instruction tuning tasks on the performance of our model, we conducted ablation studies on the ChartQA dataset using our top performing model, ChartInstruct-Flan-T5-XL (Pipeline System). These ablation experiments involved the removal of one task at a time, except for reasoning tasks, which were grouped and removed together. Due to computational constraints, we finetuned our model on the instruction tuning data for only 1 epoch only, as opposed to the 3 epochs used in the primary experiment. All other hyperparameters remained consistent with those detailed in our main experiments, as outlined in Table 5, including the fine-tuning experiments on the downstream

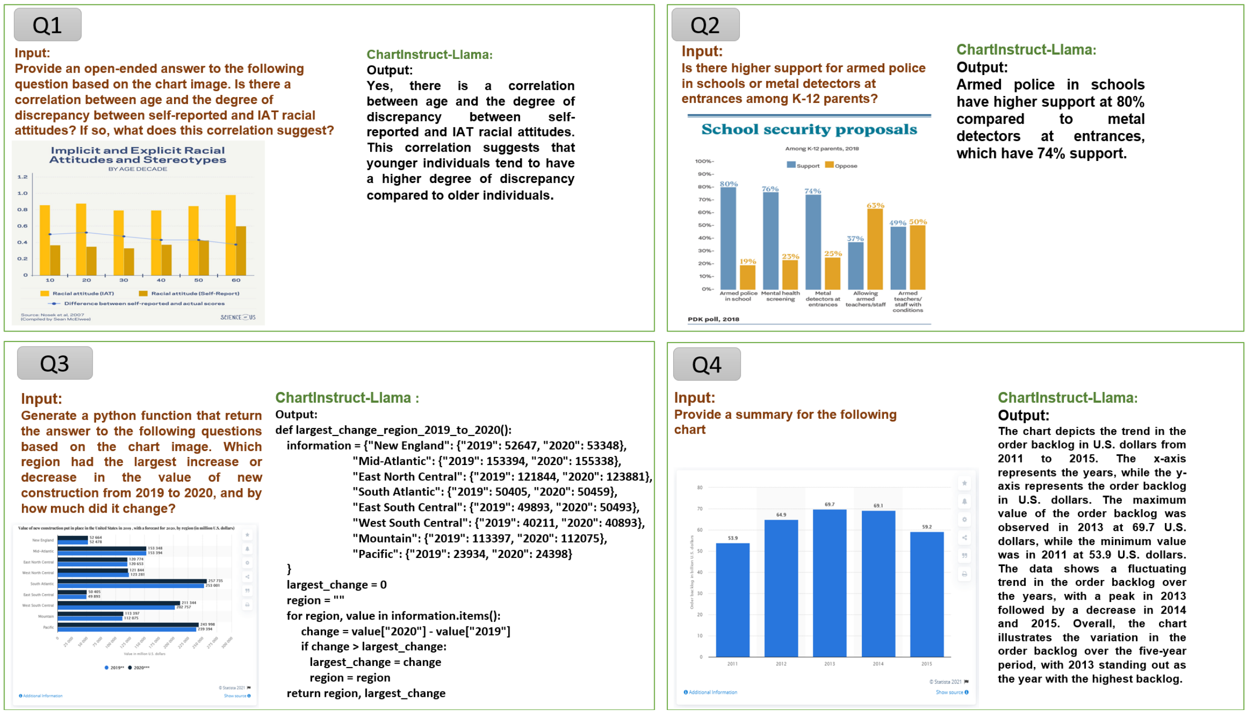


Figure 12: Sample outputs generated by ChartInstruct-Llama2 over various tasks such as LLM induced Data Correlation, Open-EndedCQA, Coding, and Chart Summarization

tasks. As depicted in Table 7, removing tasks like Chart Summarization and Open-ended Question Answering had a negligible effect on the performance on ChartQA. However, a more significant performance decline was observed upon the exclusion of the fact-checking task, which is important for enhancing the models data retrieving and reasoning capabilities. This decline was further amplified when reasoning-associated tasks (CoT and Coding) were removed, underscoring their critical role in improving the numerical reasoning capabilities of our model.

Model	ChartQA (RA)
ChartInstruct-Flan-T5-XL	70.08
No Open-ended Question Answering	69.68
No Chart Summarization	69.76
No Fact Checking	66.28
No Novel Tasks	69.20
No CoT Reasoning/Coding	63.36

Table 7: ChartInstruct ablations on ChartQA benchmark.

Rules:

Below you will see a series of charts, each chart accompanies one question/task and two models' answers. Your goal is to rate the quality of answers based on the following criteria.

(i) Informativeness: how much information from the chart does the answer cover? Ideally, an informative response should contain high-level insights from the chart
(ii) Relevance: how relevant the answer is to the input task?
(ii) Factual Correctness: how factually correct is the answer (facts mentioned are supported by the chart);

Please rate from 1(worst rating) to 5 (best rating).

Figure 13: Evaluation Rules of the human study for the three following metrics: informativeness, relevance, and factual correctness.

5.3 Human Evaluation on Chart Tasks

Reference-based evaluation metrics like BLEU-score may not align with human-perceived text quality attributes [69, 103]. To ensure accurate evaluation of our approach, we conducted a human experiment, assessing the generated responses from UniChart and our ChartInstruct-Llama2 model across three metrics: (a) Informativeness, (b) Relevance, and (c) Factual Correctness.

For the study, we chose 150 samples that are unseen by both UniChart and ChartInstruct-Llama2. Half of them are randomly from the ChartQA test set,

	Informativeness	Relevance	Factual
UniChart[75]	3.2	2.74	2.756
ChartInstruct-Llama2	3.848	4.06	3.664
$p - value$	7.43×10^{-4}	4.42×10^{-5}	1.31×10^{-8}

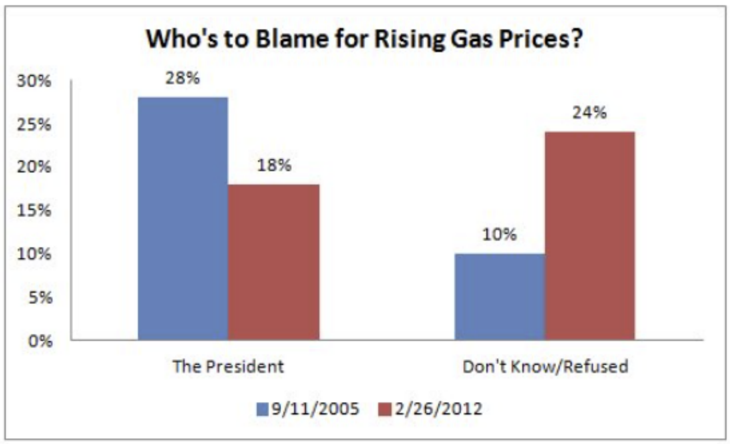
Table 8: Human evaluation results for comparing between the outputs of UniChart and ChartInstruct-Llama2. The first two rows show the average of samples across each metric. The last row shows the p-values resulted from performing Mann-Whitney U Tests.

while the other half is from a small set of web-crawled charts not used in the instruction generation pipeline. These samples contain queries from Open-ended QA, Chart Summarization, and novel instruction samples that involve a diverse set of tasks for evaluation. In terms of task distribution, 75 (50%) of the study samples belonged to novel tasks, while the other half comprised Chart-to-Text and OpenCQA tasks (40 and 35 samples). We use UniChart and ChartInstruct-Llama2 to generate responses for these samples. We asked 2 different annotators to rate the sample's responses based on the mentioned factors from 1-5, having 100 samples in common to measure their agreement level toward the responses. We presented the responses randomly to prevent any biases toward models. You can see an example from the evaluation enviornment in Figure 14.

To analyze their agreement toward the responses In order to measure that, we calculated Krippendorff's alpha[55] and found an acceptable level with the alpha coefficient of 0.447 For Factual correctness, annotators read the responses thoroughly to find out whether they involve any factual errors.

From Table 8, we observe that ChartInstruct-Llama2 significantly outperforms UniChart across all three measures of human evaluation, especially in relevance

Sample 44)



Input:
Explain the significance of the 2005 data point for 'The President' group in the context of the chart.

Output 1:
The 2005 data point for 'The President' group indicates that 28% of the population blamed the president for rising gas prices in the year 2005.

Output 2:
Overall, the data suggests that the majority of the data point for 'The President' group in the 2005 data point.

Evaluation:

Output 1
Informativeness: 4
Relevance:5
Factual Correctness:5

Output 2
Informativeness: 4
Relevance: 3
Factual Correctness: 2

Figure 14: An example of the human study evaluation setup.

(4.06 vs. 2.74). Upon manual examination, we observed that UniChart often provides a general summary of the chart without addressing specific task instructions (sometimes repeating the same tokens), particularly evident in novel and

OpenCQA task samples. In contrast, ChartInstruct-Llama2 consistently offers relevant answers for these cases (see an example in Figure 15). Overall, these findings affirm that our instruction-tuning approach enhances the model's ability to adhere to task instructions, thereby expanding its capacity to address a wide array of new real-world chart-related scenarios beyond the capabilities of the state-of-the-art pre-trained model for the chart domain.

5.4 Error Analysis and Challenges

We reviewed our model's results across various samples to highlight the challenging aspects encountered.

Value Estimation and Comparison Charts with crowded or minimal details pose challenges in pairing visual elements (e.g., bars) with their associated values, estimating data values, and making comparisons based on visual attributes (compare based on height). For instance, errors occurred in Q1 and Q2 of Figure 16, where the correct value associated with specified items was not identified.

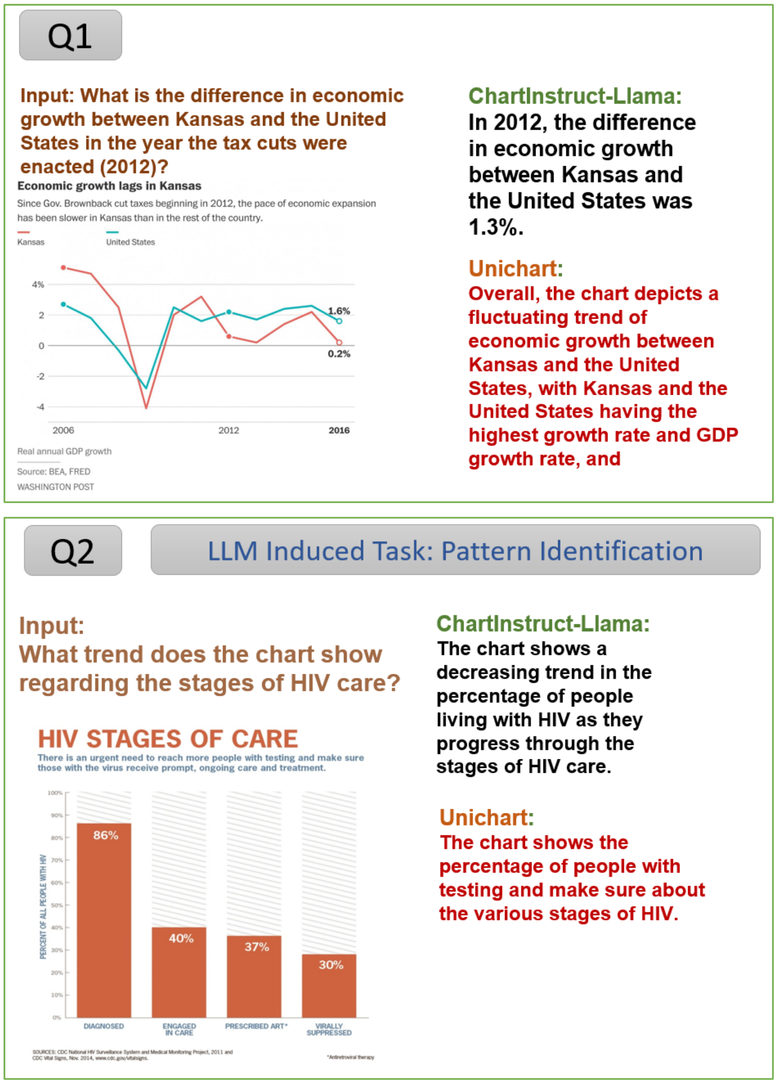


Figure 15: Comparison of ChartInstruct-Llama2 and UniChart over two WebChart novel task samples

Factual Errors Although our models have shown improved text generation quality and better utilization of available information, they still produce statements unsupported by the chart or factually incorrect. In Q3 of Figure 16, the model produces coherent text but also introduces factual errors.

Figure 16: Some samples that our ChartInstruct-Llama2 and UniChart had errors in their outputs. The errors are shown in red. UniChart often does not follow the queries and occasionally repeats tokens. On the other hand, ChartInstruct-Llama2 tends to adhere better to the instructions but occasionally encounters factual and other errors.

Numerical Reasoning Despite advancements, LLMs sometimes struggle with dependable mathematical operations [32, 66]. While achieving state-of-the-art performance in ChartQA [73] and attempting to teach the model to use external tools, LLMs still exhibit inconsistencies in calculations. Q4 in Figure 16 illustrates the unreliability of LLMs in some numerical reasoning tasks regardless of difficulty.

Figure 16 shows a few samples which our model faced a challenge generating factual and accurate responses for. In samples Q1 and Q2, our model fails to find the right value for the expected target either by estimation, matching or comparison to other visual elements. In Q3, although it generates a cohesive summary, it produces some statements that are not true. Q4 shows a numerical error that ChartInstruct-Llama didnt perform the subtraction operation correctly

6 Conclusion, Limitations and Future Work

6.1 Conclusion

We present ChartInstruct, an automatically generated dataset of chart-related instructions and two instruction systems designed for a broad range of chart-related tasks. To the best of our knowledge, this is the first instruction tuned dataset that not only includes pre-defined tasks but also many new types of tasks automatically distilled by LLMs. Our model sets the state-of-the-art performance on four different downstream tasks on various automatic measures while the human evaluation further confirms the effectiveness of our approach on many new kinds of tasks. We believe that our models and instruction-tuning dataset will be valuable resources for future research and encourage further exploration into the unique problem domain of chart understanding and reasoning.

6.2 Limitations and Future Work

First, while our research covers key tasks such as Chart Summarization, Chart Question-Answering, Open-ended Chart Question-Answering, and Chart Fact Checking, it does not cover other tasks, e.g., chart-to-table. Second, while our manual inspection of instruction-tuning dataset suggests that the novel tasks distilled by LLM are generally valid and answerable, occasionally the outputs are incorrect which may influence the instruction-tuning process. Third, although our instruction tuning approach significantly enhances the models ability to follow instructions compared to its counterpart, it does not entirely prevent the model from deviating from instructions. Fourth, despite the state-of-theart performance on the numerical reasoning task, ChartQA, our model still struggle with complex numerical questions. Finally, the model may produce factually incorrect statements in the text generation tasks.

References

[1] Gpt-4 technical report. https://openai.com/research/gpt-4. Accessed: 2023.

[2] Organisation for economic co-operation and development (oecd). https://www.oecd.org. Accessed: Jan 2024.

[3] Our world in data (owid). https://ourworldindata.org/. Accessed: Jan 2024.

[4] Pew research center. https://www.pewresearch.org/. Accessed: Jan 2024.

[5] Statista. https://www.statista.com/. Accessed: 2024.

[6] Vicuna. https://lmsys.org/blog/2023-03-30-vicuna/. Accessed: 2023.

[7] Alpaca. https://crfm.stanford.edu/2023/03/13/alpaca.html, 2023.

[8] ChatGPT. https://chat.openai.com/, 2023.

[9] Sara Abdali. Multi-modal misinformation detection: Approaches, challenges and opportunities. *arXiv preprint arXiv:2203.13883*, 2022.

[10] Mubashara Akhtar, Oana Cocarascu, and Elena Simperl. Reading and reasoning over chart images for evidence-based automated fact-checking. *arXiv preprint arXiv:2301.11843*, 2023.

[11] Mubashara Akhtar, Nikesh Subedi, Vivek Gupta, Sahar Tahmasebi, Oana Cocarascu, and Elena Simperl. Chartcheck: An evidence-based fact-checking dataset over real-world chart images. *arXiv preprint arXiv:2311.07453*, 2023.

[12] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*, 2023.

[13] Berkeley. Berkeley neural parser, 2024.

[14] Efrat Blaier, Itzik Malkiel, and Lior Wolf. Caption enriched samples for improving hateful memes detection. *arXiv preprint arXiv:2109.10649*, 2021.

[15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[16] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger,

Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[17] Juan Cao, Peng Qi, Qiang Sheng, Tianyun Yang, Junbo Guo, and Jintao Li. Exploring the role of visual content in fake news detection. *Disinformation, Misinformation, and Fake News in Social Media: Emerging Research Challenges and Opportunities*, pages 141–161, 2020.

[18] Chengliang Chai, Guoliang Li, Ju Fan, and Yuyu Luo. Crowdchart: Crowdsourced data extraction from visualization charts. *IEEE Transactions on Knowledge and Data Engineering*, 33(11):3537–3549, 2020.

[19] Ritwick Chaudhry, Sumit Shekhar, Utkarsh Gupta, Pranav Maneriker, Prann Bansal, and Ajay Joshi. Leaf-qa: Locate, encode & attend for figure question answering. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3512–3521, 2020.

[20] Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*, 2019.

[21] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer, 2020.

[22] Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. Unifying vision-and-language tasks via text generation. In *International Conference on Machine Learning*, pages 1931–1942. PMLR, 2021.

[23] J. Choi, Sanghun Jung, Deok Gun Park, J. Choo, and N. Elmqvist. Visualizing for the nonvisual: Enabling the visually impaired to use visualization. *Computer Graphics Forum*, 38, 2019.

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

[25] Komal Dadhich, Siri Chandana Daggubati, and Jaya Sreevalsan-Nair. Barchartanalyzer: Digitizing images of bar charts. In *IMPROVE*, pages 17–28, 2021.

[26] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.

[27] Seniz Demir, Sandra Carberry, and Kathleen F McCoy. Summarizing information graphics textually. *Computational Linguistics*, 38(3):527–574, 2012.

[28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[29] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[30] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[31] Yifan Du, Zikang Liu, Junyi Li, and Wayne Xin Zhao. A survey of vision-language pre-trained models. *arXiv preprint arXiv:2202.10936*, 2022.

[32] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*, 2022.

[33] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR, 2023.

[34] Li Gong, Josep M Crego, and Jean Senellart. Enhanced transformer model for data-to-text generation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 148–156, 2019.

[35] Yucheng Han, Chi Zhang, Xin Chen, Xu Yang, Zhibin Wang, Gang Yu, Bin Fu, and Hanwang Zhang. Chartllama: A multimodal llm for chart understanding and generation, 2023.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[37] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. TaPas: Weakly supervised table parsing via

pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online, July 2020. Association for Computational Linguistics.

[38] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*, 2022.

[39] Enamul Hoque and Maneesh Agrawala. Searching the visual style and structure of d3 visualizations. *IEEE transactions on visualization and computer graphics*, 26(1):1236–1245, 2019.

[40] Enamul Hoque, Parsa Kavehzadeh, and Ahmed Masry. Chart question answering: State of the art and future directions. *Journal of Computer Graphics Forum (Proc. EuroVis)*, pages 555–572, 2022.

[41] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. Applying pragmatics principles for interaction with visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):309–318, 2018.

[42] Ting-Yao Hsu, C Lee Giles, and Ting-Hao'Kenneth' Huang. Scicap: Generating captions for scientific figures. *arXiv preprint arXiv:2110.11624*, 2021.

[43] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In

International conference on machine learning, pages 4904–4916. PMLR, 2021.

[44] Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2018.

[45] Kushal Kafle, Robik Shrestha, Brian Price, Scott Cohen, and Christopher Kanan. Answering questions about data visualizations using efficient bimodal fusion. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1487–1496, 2020.

[46] Shankar Kanthara, Rixie Tiffany Ko Leong, Xiang Lin, Ahmed Masry, Megh Thakkar, Enamul Hoque, and Shafiq Joty. Chart-to-text: A large-scale benchmark for chart summarization. *arXiv preprint arXiv:2203.06486*, 2022.

[47] Shankar Kantharaj, Xuan Long Do, Rixie Tiffany Leong, Jia Qing Tan, Enamul Hoque, and Shafiq Joty. OpenCQA: Open-ended question answering with charts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11817–11837, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[48] Shankar Kantharaj, Xuan Long Do, Rixie Tiffany Ko Leong, Jia Qing Tan, Enamul Hoque, and Shafiq Joty. Opencqa: Open-ended question answering with charts. In *Proceedings of EMNLP (to appear)*, 2022.

[49] Shankar Kantharaj, Xuan Long Do, Rixie Tiffany Ko Leong, Jia Qing Tan, Enamul Hoque, and Shafiq Joty. Opencqa: Open-ended question answering with charts. *arXiv preprint arXiv:2210.06628*, 2022.

[50] Shankar Kantharaj, Rixie Tiffany Leong, Xiang Lin, Ahmed Masry, Megh Thakkar, Enamul Hoque, and Shafiq Joty. Chart-to-text: A large-scale benchmark for chart summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4005–4023, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[51] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in neural information processing systems*, 33:2611–2624, 2020.

[52] Dae Hyun Kim, Enamul Hoque, and Maneesh Agrawala. Answering questions about charts and generating visual explanations. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–13, 2020.

[53] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517. Springer, 2022.

[54] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR, 2021.

[55] Klaus Krippendorff. Computing krippendorff's alpha-reliability, 2011.

[56] Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. *arXiv preprint arXiv:2210.03347*, 2022.

[57] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*, pages 18893–18912. PMLR, 2023.

[58] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[59] Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. Llava-med: Training a large language-and-vision assistant for biomedicine in one day. *arXiv preprint arXiv:2306.00890*, 2023.

[60] Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11336–11344, 2020.

[61] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[62] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[63] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

[64] Can Liu, Yun Han, Ruike Jiang, and Xiaoru Yuan. Advisor: Automatic visualization answer for natural-language question on tabular data. In

2021 IEEE 14th Pacific Visualization Symposium (PacificVis), pages 11–20, 2021.

[65] Fangyu Liu, Julian Martin Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. Deplot: One-shot visual language reasoning by plot-to-table translation. *arXiv preprint arXiv:2212.10505*, 2022.

[66] Fangyu Liu, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Yasemin Altun, Nigel Collier, and Julian Martin Eisenschlos. Matcha: Enhancing visual language pretraining with math reasoning and chart derendering. *arXiv preprint arXiv:2212.09662*, 2022.

[67] Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. Mmc: Advancing multimodal chart understanding with large-scale instruction tuning. *arXiv preprint arXiv:2311.10774*, 2023.

[68] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[69] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment, 2023.

[70] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer

using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[71] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[72] Junyu Luo, Zekun Li, Jinpeng Wang, and Chin-Yew Lin. Chartocr: Data extraction from charts images via a deep hybrid framework. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1917–1925, 2021.

[73] Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2263–2279, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[74] Ahmed Masry and Enamul Hoque. Integrating image data extraction and table parsing methods for chart question answering. *Chart Question Answering Workshop, in conjunction with the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–5, 2021.

[75] Ahmed Masry, Parsa Kavehzadeh, Xuan Long Do, Enamul Hoque, and Shafiq Joty. Unichart: A universal vision-language pretrained model for

chart comprehension and reasoning. *arXiv preprint arXiv:2305.14761*, 2023.

[76] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022.

[77] Fanqing Meng, Wenqi Shao, Quanfeng Lu, Peng Gao, Kaipeng Zhang, Yu Qiao, and Ping Luo. Chartassisstant: A universal chart multimodal language model via chart-to-table pre-training and multitask instruction tuning. *arXiv preprint arXiv:2401.02384*, 2024.

[78] Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. Plotqa: Reasoning over scientific plots. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1527–1536, 2020.

[79] Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. Automated fact-checking for assisting human fact-checkers. *arXiv preprint arXiv:2103.07769*, 2021.

[80] Arpit Narechania, Arjun Srinivasan, and John Stasko. Nl4dv: A toolkit for generating analytic specifications for data visualization from natural language queries. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):369–379, 2021.

[81] Jason Obeid and Enamul Hoque. Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 138–147, Dublin, Ireland, December 2020. Association for Computational Linguistics.

[82] Jason Obeid and Enamul Hoque. Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 138–147. Association for Computational Linguistics, 2020.

[83] Jason Obeid and Enamul Hoque. Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model, 2020.

[84] Jason Obeid and Enamul Hoque. Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model. *arXiv preprint arXiv:2010.09142*, 2020.

[85] OpenAI. GPT-4 Technical Report, 2023.

[86] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[87] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022.

[88] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[89] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

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

[91] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[92] Raian Rahman, Rizvi Hasan, Abdullah Al Farhad, Md Tahmid Rahman Laskar, Md Hamjajul Ashmafee, and Abu Raihan Mostofa Kamal. Chartsumm: A comprehensive benchmark for automatic chart summarization of long and short summaries. *arXiv e-prints*, pages arXiv–2304, 2023.

[93] Revanth Reddy, Rahul Ramesh, Ameet Deshpande, and Mitesh M. Khapra. Figurenet : A deep learning model for question-answering on scientific plots. *Proceedings of the International Joint Conference on Neural Networks*, 2019-July, 2019.

[94] Ehud Reiter. An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 97–104. Association for Computational Linguistics, 2007.

[95] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[96] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023.

[97] Vidya Setlur, Sarah E. Battersby, Melanie Tory, Rich Gossweiler, and Angel X. Chang. Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST 2016, pages 365–377, New York, NY, USA, 2016. ACM.

[98] Vidya Setlur, Enamul Hoque, Dae Hyun Kim, and Angel X. Chang. Sneak pique: Exploring autocompletion as a data discovery scaffold for supporting visual analysis. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 966978, New York, NY, USA, 2020. Association for Computing Machinery.

[99] Kantharaj Shankar, Leong Rixie Tiffany Ko, Lin Xiang, Masry Ahmed, Thakkar Megh, Hoque Enamul, and Joty Shafiq. Chart-to-text: A large-scale benchmark for chart summarization. In *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2022*, 2022.

[100] Shivam Sharma, Firoj Alam, Md Shad Akhtar, Dimitar Dimitrov, Giovanni Da San Martino, Hamed Firooz, Alon Halevy, Fabrizio Silvestri, Preslav Nakov, and Tanmoy Chakraborty. Detecting and understanding harmful memes: A survey. *arXiv preprint arXiv:2205.04274*, 2022.

[101] Danqing Shi, Xinyue Xu, Fuling Sun, Yang Shi, and Nan Cao. Calliope: Automatic visual data story generation from a spreadsheet. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):453–463, 2020.

[102] Hrituraj Singh and Sumit Shekhar. Stl-cqa: Structure-based transformers with localization and encoding for chart question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3275–3284, 2020.

[103] Aaron Smith, Christian Hardmeier, and Joerg Tiedemann. Climbing mont BLEU: The strange world of reachable high-BLEU translations. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 269–281, 2016.

[104] Andrea Spreafico and Giuseppe Carenini. Neural data-driven captioning of time-series line charts. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pages 1–5, 2020.

[105] Jaya Sreevalsan-Nair, Komal Dadhich, and Siri Chandana Daggubati. Tensor fields for data extraction from chart images: bar charts and scatter plots. In *Topological Methods in Data Analysis and Visualization VI: Theory, Applications, and Software*, pages 219–241. Springer, 2021.

[106] Arjun Srinivasan and John Stasko. Orko: Facilitating multimodal interaction for visual exploration and analysis of networks. *IEEE transactions on visualization and computer graphics*, 24(1):511–521, 2017.

[107] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019.

[108] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

[109] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, and Jiahui Yu et al. Gemini: A family of highly capable multimodal models, 2023.

[110] Gemma Team. Gemma 2: Improving open language models at a practical size, 2024.

[111] T5 Team. Scaling instruction-finetuned language models, 2022.

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

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

[114] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy

Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[115] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[116] Peifang Wang, Olga Golovneva, Armen Aghajanyan, Xiang Ren, Muhao Chen, Asli Celikyilmaz, and Maryam Fazel-Zarandi. Domino: A dual-system for multi-step visual language reasoning. *arXiv preprint arXiv:2310.02804*, 2023.

[117] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Align-

ing language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*, 2023.

[118] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. *arXiv preprint arXiv:2108.10904*, 2021.

[119] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[120] Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2733–2743, 2023.

[121] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, Chenliang Li, Yuanhong Xu, Hehong Chen, Junfeng Tian, Qian Qi, Ji Zhang, and Fei Huang. mplug-owl: Modularization empowers large language models with multimodality, 2023.

[122] Fangfang Zhou, Yong Zhao, Wenjiang Chen, Yijing Tan, Yaqi Xu, Yi Chen, Chao Liu, and Ying Zhao. Reverse-engineering bar charts using neural networks. *Journal of Visualization*, 24:419–435, 2021.

[123] Mingyang Zhou, Yi Fung, Long Chen, Christopher Thomas, Heng Ji, and Shih-Fu Chang. Enhanced chart understanding via visual language pre-training on plot table pairs. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1314–1326, Toronto, Canada, July 2023. Association for Computational Linguistics.

[124] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[125] Jiawen Zhu, Jinye Ran, Roy Ka-wei Lee, Kenny Choo, and Zhi Li. Autochart: A dataset for chart-to-text generation task. *arXiv preprint arXiv:2108.06897*, 2021.

[126] Jialong Zou, Guoli Wu, Taofeng Xue, and Qingfeng Wu. An affinity-driven relation network for figure question answering. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2020.

Appendices

A Task Prompts

A.1 Gemini Data Table Generation Prompt

We used the following prompt concatenated with chart image for data table and metadata generation using Gemini-Vision Pro.
Prompts: Try to come up with a JSON that represents the underlying data table and other metadata for the input chart image. Do not generate anything besides the JSON object.

A.2 Chain-of-Thought Reasoning

Prompt: For this task, we considered two types of question, one of which requires tracking variables to generate the final answer, while the other one doesn't need it. As a result, we came up with different prompt for each of them. We generated the instructions by feeding these prompts concatenated with chart metadata to GPT-4.

A.2.1 Variable-based Prompt

Prompt: First of all, know that you must act like you can't do basic match even addition or subtraction. However, you might feel a need for calculation in your incoming tasks. In that case, you can tell me to calculate by calling $Calculator(X =$

$OPERATION)$. <OPERATION> can be executable python code and X is a variable. Use the defined variables in the next sentences when needed like a python code. Do not define the variables in the text. Use this format "$DEFINE(variable_name = X)$" where X is the value of the variable. Youll be given a datatable in json string format which is an underlying datatable for a chart. Although you cant see the chart, you should mention any references that come up to the chart instead of the table Your responses must be limited to question and answer pairs related to the data, strictly avoiding any conversational language or fillers. You have to come up with six pairs of question and answers that ask for values that require calculations such as differences, totals, and statistical measures such as median, mode, and mean. To come up with your answer, you have to break down your solution into several basic steps. In each step, you explain what are you going to do, and if that is a mathematical operation, you have to mention the formula and fill it with the respected numbers. If you want to list some numbers, do it when there are less than 8 of them. Since you are unable to do calculations, you may return one of the defined variables from your previous steps as your final answer. Consider the following template: " The Answer is <ANSWER>." to wrap up your generated steps, replace <ANSWER> in the mentioned template with your final answer word without ANY explanations or assignment. Please return the list of questions and responses in a json format. I should be able to parse it. Each dictionary must contain 3 field 'question', 'steps', 'answer' filled with the requested formation. The answer section must only involve a python variable previously defined. No numbers should appear in this section.

A.2.2 Non-Variable-based Prompt

Prompt: Youll be given a datatable which is an underlying datatable for a chart. Although you cansee the chart, you should mention any references that come up to the chart instead of the table Your responses must be limited to question and answer pairs related to the data, strictly avoiding any conversational language or fillers. You have to come up with six pairs of question and answers that ask for achart value, or comparison not asking for a number, and a general trend. There must be at least two retrieval questions and one general trend. To come up with your answer, you have to break down your solution into several basic steps. In each step, you explain what are you going to do, and if that is a mathematical operation, you have to mention the formula and fill it with the respected numbers. If you are comparing two or several numbers retrieved from the table, you must mention them. Consider the following template: " The Answer is <ANSWER>." to wrap up your generated steps, replace <ANSWER> in the mentioned template with your final answer word without ANY explanations or assignment. Please return the list of questions and responses in a json format. I should be able to parse it. Each dictionary must contain 3 field 'question', 'steps', 'answer' filled with the requested formation.

A.3 Chart Summarization Prompt

We used the following prompt for chart summarization concatenated with the datatable and other metadata of a chart image using Chat-GPT3.5-turbo.

Prompt: Summarize a chart based on a provided data table, highlighting key aspects such as data background, maximum and minimum values, value comparisons, and trends across categories. The summary should not directly mention the input table, instead referring to the 'chart' or 'chart elements'. You'll also be given the title of the chart to get a context on what is shown in the chart. Do not use the title as it is like "the chart is titles TITLE", always try to infersomething new based on it. Include a brief explanation of the elements represented on different axises(if applicable). The entire summary should be cohesive, in paragraph form without bullet points, and not exceed 150 words. Conclude with a final sentence that encapsulates the most significant findings from the chart.

A.4 Open-ended Question Answering Prompt

We used the following prompt for Open-ended Question Answering concatenated with the datatable and other metadata of a chart image using chat-gpt3.5-turbo.

Prompt: Generate question-answer pairs based on a data table representing a chart. You will also be given the title to have additional context. The questions should vary in complexity, ranging from simple 'what', 'which', 'when' questions to more challenging 'how' and 'why' questions. Refer to 'chart' or 'chart elements' instead of the data table. Ensure the answers are cohesive and fluent, using diverse vocabulary, with each answer spanning 3 to 5 sentences. The questions should encompass different aspects related to a chart figure. Each turn of conversation should consist of 3 or 4 question-answer pairs, covering a range of insights, explanations, and complexities as mentioned. There's no specific order for the difficulty of the questions. Put ! before starting each question, and * before each answer.

A.5 Fact Checking

We used the following prompt for fact checking concatenated with the datatable and other metadata of a chart image using chat-gpt3.5-turbo.

Analyze a chart and generate questions about its data, forming pairs of statement and answer. Half of the questions should be supported by the chart's data, while the other half are refuted. Avoid using terms like 'rows', 'columns', or 'elements' from the data table; refer to 'chart' or 'chart image' instead. Each response must state whether it 'supports' or 'refutes' the question, followed by a brief explanation. The questions should cover comparisons of values or trends, basic statistical values (maximum, minimum, mean, median, mode) without using exact numbers from the chart. Ensure a diverse range of questions addressing various visual aspects of the chart, resulting in 4-6 turns of Questions and Answers. Put ! before starting each question, and * before each answer.

A.6 Code Generation Prompt

We used the following prompt for Python code generation concatenated with the datatable and other metadata of a chart image using GPT-4.

Prompt: Assume that you are an expert in solving problems by implementing the script text of a python funtion to solve that problem. You are expected to find the answer to an input question step-by-step sharing explanations as comments in-between the python commands. Right now, the input doesnt involve a question, so you have to come up with a question as well. The question can ask one item like an finding an element or a statistical measure like mean, median, etc. After finishing the solution, return the pyhton script. The requirements for the outputs are: The output MUST BE ONLY a piece of python text script without any lingual explanation in ENGLISH. Don't use any indication that you are using the table. Try to make the sentence in a way that you are looking at a chart instead of a table. The output must only be a generated question in the form of a question at the beginning outside of the python script and then, a python function solving that question with no other commands or explanations in the body of the script. It takes no input arguments. Every command including the definition of data variables must be inside that function, and the function returns the answer variable. The function must not be called. These questions should vary in complexity: easy (value retrieval), medium (basic calculations or single comparisons), and hard (compositional questions involving more than one math operation like summation, Subtraction, division, Multiplication, and comparison on different metrics such as maximum, minimum, median or finding a complex trends, like asking for the difference between median, maximum, mean, minimum, and etc.). For each table, come up with three rounds of distinct Questions and Answers pairs with three hard compositional questions. Put ! before starting each question, and * before each answer.

A.7 Novel Tasks Prompt

We used the following prompt for novel tasks concatenated with the datatable and other metadata of a chart image using GPT-4.

Prompt: Generate different instruction tuning tasks for an LLM that we are trying to tune for Chart Understanding. Your response should be be in a JSON format where each example has three fields: task type, input, and expected output. Use the following chart data to generate 10 unique tasks that do not have overlap with two specific tasks: Summarizing the Chart, and specific information retrieval. Do not generate anything besides the JSON object.