# Chart-to-Table Conversion: A Survey

Mohamed Fayed, Kruthik Ravikanti, Bruce Walker
mohamed.fayed@gatech.edu, kravikanti3@gatech.edu,
bruce.walker@psych.gatech.edu

***Abstract***—Multimodal Large Language Models (MLLMs) have shown impressive visual capabilities in many Visual Question Answering tasks. Chart-to-Table Conversion the problem of extracting structured tabular data out of input images, enabling better accessibility and higher storage efficiency. In this survey, we aim to review the recent advancements in Chart-to-Table task, analyze existing datasets and benchmark some state-of-the-art models on this task. Our quantitative and qualitative analysis highlights strengths and limitations of those approaches in handling complex charts, preserving numerical precision and generalize across various types of charts. Our study reveals significant room for improvement in Chart-to-Table Conversion, emphasizing the need for more specialized and robust methods.

## 1 INTRODUCTION

Data visualization is an essential tool for communicating information efficiently, with charts and graphs playing a fundamental role in various domains, including finance, healthcare, and scientific research. Those visualizations are usually saved in form of images, which are consuming a significant amount of disk storage. However, extracting structured tabular data from chart images remains a challenging problem, particularly for automated systems. This challenge is especially critical for accessibility, where individuals with visual impairments rely on machine-readable formats to interpret visual information.

Chart-to-Table is the task of extracting data points from an image of a chart into a table usually in markdownLiu et al., Masry et al., 2022a, 2024b. This task was usually tackled via combination of two or more approaches from rule based tecniques, image processing, CNN based deep learning architectures and optical character recognition (OCR). Those techniques struggle with diverse chart formats, occlusions and variations in visual styles. More recently, this task was tackled via transformer based models and Multimodal LArge Language Models (MLLMs), which provide more robust solutions.

In this survey, we provide a comprehensive review of Chart-to-Table conversion techniques, covering benchmark datasets, model architectures, evaluation metrics, and existing challenges. Furthermore, we assess the performance of several state-of-the-art transformers based models, including MLLMs, on this task, highlighting their strengths and weaknesses. Our findings suggest that while those models exhibit promising results, significant challenges remain, including generalization across different chart types, handling visual input effectively, and improving numerical fidelity.

Our main contributions are:

1. Survey recent advancements in Chart-to-Table task, to the best of our knowledge, this is the first survey focusing on Chart-to-Table Conversion,
2. do comprehensive qualitative and quantitative analysis on various benchmarks covering various common types of charts, and
3. highlight strengths and limitations of those models, emphasizing on rooms for improvement in tackling Chart-to-Table Conversion.

## 2 CHART-TO-TABLE IN LITERATURE

There has been work in operating in charts in a way or another. Some researchers aim at converting charts to tables (Chart-to-Table Conversion). Others have worked on Chart Question Answering and Chart Summarization. In this work, we focus on Chart-to-Table Conversion.

Chart-to-Table Conversion requires tackling Optical Character Recognition (OCR) and comprehensive Visual Understanding. In this section, we review related work in Traditional Computer Vision, Deep Learning and Multimodal Large Language Models and how they influenced the evolution of this task.

### 2.1 Computer Vision and Deep Learning

It all begins with computer vision (CV)-based methods. Early attempts to extract tabular data from charts utilized traditional image processing techniques like segmentation and edge detection that could identify visual elements like bars, axes, and legends Sreevalsan-Nair, Dadhich, and Daggubati, 2021. These approaches were effective to some extent, but they struggled with the many variations found in chart styles and occlusions. Furthermore, rule-based approaches were often unsuccessful due to complex charts that had overlapping elements or non-standard layouts Poco and Heer, 2017. As a result, researchers pursued more robust techniques that could better generalize across diverse chart designs.

Fast forward, researchers gave deep learning a try. Systems such as ChartSense Jung et al., 2017 and ChartOCR Luo et al., 2021 utilized convolutional neural networks (CNNs) to classify types of charts and extract available data with higher accuracies. For instance, ChartOCR combined rule-based methods with deep CNNs to balance generalization and accuracy, achieving strong performance across multiple chart types Luo et al., 2021. However, these methods needed extensive labeled datasets and lacked generalization across different chart formats.

There were many efforts towards handling specific types of charts using either Computer Vision or Deep Learning tecniques. ChartParser Kumar, Ganu, and Guha, 2022 was a pipeline that focuses on extracting data points for various kinds of bar charts. They build rule based algorithm using OCR and image processing techniques. Another group of researchers Li et al., 2022 focused on extracting underlying data points for camera photos of line charts using custom architecture based on Faster-RCNN Ren et al., 2015. They gathered synthetic and real datasets for line charts to improve the performance of their model. Their approach shows promising direction in utilizing synthetic data in improving models on real world test images. This direction of research highlights challenges of each kind of chart, but it has many disconnected componenets and their results are not necessarily comparable.

### 2.2 Pretraining and Fine-tuning

After the introduction of transformers Vaswani et al., 2017, Pretraining and Fine-tuning paradigm caught researchers attention. Masry et al., 2023 gathered a large dataset of 6.9M questions and charts, and used it to pre-train UniChart model. Another group of researchers Liu et al., 2022b introduced Matcha model, which was a fine-tuned version of Pix2Struct Lee et al., 2023 on many tasks including Chart-to-Table. Later on, Liu et al., 2022a continued pre-training it on Chart-to-Table only to create Deplot. They forwarded the generated table, human query and an example to FlanPaLM 540B Chung et al., 2024 to answer complex queries. Cheng, Dai, and Hauptmann, 2023 took a different approach by training a transformers based chart component detection and combine it with extended pretrained T5 Raffel et al., 2020 or TaPas Herzig et al., 2020 models.

### 2.3 Multimodal Large Language Models (MLLMs)

MLLMs has made significant progress in many tasks and Chart-to-Table was no exception. One direction is about utilizing general domain LLMs without any tuning. This direction includes Prompting and Retrieval Augmented Generation techniques Cao et al., Voigt et al., 2024, 2023 to improve the capabilities of general LLMs on chart related tasks.

Another direction is to fine-tune LLMs on Chart-specific Instruction following datasets. Masry et al., 2024a introduced an instruction following dataset and instruction tuned both LLaMA 2 7B Touvron et al., 2023 and Flan-T5 XL 3B Chung et al., 2024 on it. This instruction tuning strategy was proved to make a generalized model that can handle unseen tasks. Similarly, Masry et al., 2024b fine-tuned PaliGemma Beyer et al., 2024 to create a 3B ChartGemma. Another key distinguishing contribution was the method of instruction following data generation. They generated for predefined tasks, such as Chain of Thought and Chart-to-Tables in form of markdown, and open-ended tasks, such as justifying trends in charts and describing visual elements.

A unique direction is chart-to-code generation, which aims to provide lossless representations of charts. Matcha Liu et al., 2022b was the first model to include Chart-to-Code within its training tasks. ChartCoder Zhao et al., 2025 is a new model that translates chart images into executable Python code (e.g., using matplotlib). Trained on the newly released Chart2Code-160k dataset containing 160k chart-code pairs across 27 chart types, ChartCoder uses a Code LLM backbone and a Snippet-of-Thought prompting method for step-by-step generation. While intended for re-rendering, the output code preserves all chart data, making it a viable route for Chart-to-Table extraction through code interpretation.

Another recent model, ChartCitor Goswami et al., 2025, focuses on improving how chart question-answering models support their answers with evidence. Instead of just generating answers. ChartCitor uses a group of LLM agents to first extract a table from a chart, then break the answer into smaller claims, and finally find the exact table cells that support those claims. It also connects these cells back to specific parts of the chart using object detection and visual prompts. ChartCitor does not aim to extract full tables, it introduces a useful method for linking visual data to structured output and improves transparency in chart understanding tasks, which makes it a good candidate for verifying correctness of generated tables.

### 2.4 Chart-to-Table Evaluation

There has been many metrics for evaluating Chart-to-Table conversion. In Luo et al., 2021, they introduced different metric for each kind of chart:

1. For Bar Charts, defined a custom distance function for pairwise point comparison and find minimum cost between prediction and ground truth,
2. For Line charts, it is handled as continuous similarity problem.œœ For each predicted line, it computes the pointwise error between it and each ground truth line, and choose the minimal value.
3. For Pie Charts, they consider its scoring as sequence matching problem, thus solved using dynamic programming.

Relative Number Similarity Score Masry et al., 2022, also known as Relaxed Accuracy Measure, is about computing highest accuracy of generated numbers relative to ground truth. However, it has two main drawbacks:

1. It does not consider the position of numbers within the table, and
2. It ignores textual errors.

To overcome those limitations, Relative Mapping Similarity (RMS) Liu et al., 2022a computes edit distance between columns names, compute accuracy of values within columns of least edit distances and compute F1-score. [1] To learn more about the algorithm, please check appendix A

## 3 METHODOLOGY

### 3.1 Datasets

There has been many datasets introduced for various chart related tasks, with no special attention to Chart-to-Table Conversion. Aftersome investigation, we found that some datasets include chart images with their corresponding data points. In our analysis, we focus on reporting scores on two of those testsets: ICPR22 Rousseau and Kapralos, 2023 and PlotQA Methani et al., 2020.

***ICPR22 testset Rousseau and Kapralos, 2023***—was gathered from research papers published on PubMed Central website.[2] Those publications are in biomedical and life sciences domains. It contains 443 charts splitted into 5 types: Line Charts,

[1] our RMS implementation can be found here.

[2] https://pmc.ncbi.nlm.nih.gov

Horizontal and Vertical Bar Charts, Scatter Plot and Vertical Box Plot. It represents a source of real-world generated charts, as they were generated by a large set of researchers.

***PlotQA Methani et al., 2020*—**was made by gathering data from various online sources, such as World Bank and Open Data, generate plots out of these data points, and ask annotators questions about those provided charts. In our work, we focus on the data points used in constructing the charts only. Its test set contains 33657 charts divided equally among dotted line charts, line charts, and vertical and horizontal bar charts. We hypothesized that there is no need to test on all 33k charts, so we performed comparison among scores for many sample sizes. Table 1 shows that it is enough to compute scores on a sample of 4k randomly selected charts.

***Test-Sample:*—**For the sake of qualitative analysis, we selected a set of charts from PlotQA and ICPR22 testsets. We randomly selected from each kind of chart in PlotQA. However, for ICPR22, we selected charts from various data sizes, i.e. of various sizes of json files. We noticed that larger file sizes are directly correlated to larger count of data points of corresponding charts.

### 3.2 Models

For our study, we tested the following models in zero-shot setting:

- Gemini 1.5 Flash Team et al., 2024: A general purpose lightweight MLLM, which is larger than other models in this list.
- ChartGemma Masry et al., 2024b: A specialized model in chart summarization, question answering and reasoning about charts. It utilizes PaliGemma 4B Beyer et al., 2024 as its backbone, and was tuned on Visual Chart Instructions dataset.
- Deplot Liu et al., 2022a: A 282M parameters Vision Encoder-Text Decoder transformer architecture. It was trained on Chart-to-Table Conversion task.

### 3.3 Prompts

For deDeplot, we used the prompt recommended in its Hugging Face Model Card: "Generate underlying data table of the figure below:". For ChartGemma, we used a simple prompt of "program of thought: extract data points and formulate them into markdown table. Note: all data points should be either integer or float. Do not write percentages as ’56

For Gemini 1.5 Flash, we used a longer prompt, and if its output is not parseable into json object, we prompt it again to fix its format. The main prompt is:

```
"System Message:     You are an EXPERT in scientific visualizations analysis and ch
You should help people with disabilities in giving an overview of the chart and con
    prompt: For the following image, tell me the name of x and y axes, and the valu
    The output should be in format of
    {x_axis_name: x_axis_vavlues_list, y_xis_name: y_axis_values_list}"
```

### 3.4 Evaluation

To compare among those models, we compute Relative Mapping Similarity (RMS) Liu et al., 2022a and do qualitative analysis on our Test-Sample testset.

## 4 RESULTS AND DISCUSSION

### 4.1 Scores

In this section, we report the scores for each model. While testing them, we noticed that they have issues in generating markdowns/jsons that can be correctly parsed by pandas. So, we also report Success Rate (SR) for each model, such that $SuccessRate(SR) = countofcorrectlyparsedcharts/countofpredictedchartswithinthetestsetbyselec$

We reported the results on a subset of PlotQA charts and all of ICPR22. We were able to generate around 7000 charts from PlotQA. We hypothesized that the size of 33k test images is very large and we should not find significant difference between scores on 3k and 33k. To validate this hypothesis, we computed scores for 3k, 4k, 5k and 6k charts to see whether there is a significant need to infere on all 33k images. Additionally, we computed the scores on the entire testset using deplot model. As shown in table 1, there are small differences across samples of 4k, 5k and k relative to variations among scores of all models. This means that a sample of 4k is sufficient to provide a reliable estimate to model's performance.

### 4.2 Discussion

Fianl scores are shown in table 2. Gemini 1.5 Flash achieved the highest Success Rate (SR), indicating its strong ability to produce well-formatted and parseable outputs. However, Deplot outperformed all models in terms of Relative Mapping Similarity (RMS), showcasing its superior alignment with the ground truth table

| | Gemini 1.5 Flash | |
|---|---|---|
| Size | SR | RMS |
| 1.5k | 86.41% | 55.43% |
| 3k | 85.94% | 55.17% |
| 4k | 8606% | 55.26% |
| 5k | 86.39% | 55.56% |
| 6k | 86.40% | 55.56% |

| | ChartGemma | |
|---|---|---|
| 1.5k | 15.93% | 24.93% |
| 3k | 17.53% | 29.66% |
| 4k | 17.21% | 30.26% |
| 5k | 18.15% | 31.26% |
| 6k | 18.32% | 31.17% |
| 6946 | 18.94% | 31.24% |

| | Deplot | |
|---|---|---|
| Size | SR | RMS |
| 1.5k | 83.6% | 83.63% |
| 3k | 84.57% | 83.72% |
| 4k | 84.65% | 83.77% |
| 5k | 84.24% | 83.62% |
| 6k | 83.93% | 83.74% |
| 6946 | 83.7% | 83.66% |
| 33k (all) | 83.41% | 83.68% |

***Table 1***—Scores of all models for different sample sizes from PlotQA. The first 2 digits are the same starting at sample size of 4k, which implies we do not need more than 4k images to get scores of precision of 2 digits.

| Model | PlotQA | | ICPR22 | |
|---|---|---|---|---|
| | SR | RMS | SR | RMS |
| Gemini 1.5 Flash | 86.5% | 55.6% | 76.8% | 19% |
| ChartGemma | 18.9% | 31.2% | 48.4% | 17.2% |
| Deplot | 83.7% | 83.66% | 75.8% | 19.6% |

***Table 2***—Models scores on 7k subset of PlotQA and all ICPR22 testsets.

values. On the other hand, ChartGemma recorded the lowest performance in both SR and RMS.

The relatively high SR of Gemini 1.5 Flash suggests that a powerful text decoder plays a crucial role in generating structured and clean outputs. Meanwhile, Deplot's strong RMS score—despite being a smaller, specialized model—underscores the value of task-specific architectural tuning. These results suggest that instruction fine-tuning (as in ChartGemma) may not be sufficient to achieve high-quality Chart-to-Table conversions without deeper task alignment and robust visual reasoning.

Low scores of all models in ICPR22 testset suggests that it is difficult testset. After manual inspection, we noticed that it has many issues causing RMS scores to be low. One cause is that many line charts have no labels for their lines. Many columns in tables of the testset contain "unnamed data series 0", which has 1 normalized edit distance with whatever models generate as columns names, thus 0 RMS score occurs frequently. In a future work, we would also take into account computing RMS both with and without considering normalized edit distance. Another reason for those low scores is the failure of all models in properly extracting data from scatter plots and vertical box plots, as illustrated in section 4.4.3 and section 4.4.4 respectively.

## 4.3 Text Recognition

For the sample we analyzed, we found a single error in deplot recognizing name of second row in table 28308 in PlotQA. Otherwise, all text was correctly recognized by all models. However, tables 13, 12 and 11 show models' tendency to labelize even if there are no labels in the input image. [3] [4]

## 4.4 Values Extraction

For PlotQA and ICPR22 samples, it is frequent to find errors like:

1. rounding errors, e.g. $15.42- > 15 and 15.6- > 15$.

[3] the prediction of Gemini and Ground Truth have no labels for x-axis, but ChartGemma made years as labels.

[4] In some cases, the ground truth is mislabelled. The reference has no values for x-axis, but the image includes them as in 6.

2. Precision Errors: we have noticed that the models can not predict more than 3 digits for Gemini 1.5 Flash and ChartGemma or 4 digits for Deplot per each value, e.g. $126765000.0 -> 156000000$.
3. In case of near values, e.g. $24.18, 24.09$, there might be some errors, e.g. predicting 23 instead of 24. For that kind of error, it may result in changing trend, e.g. steady performance may seam as decreasing. [5]
4. All models can differentiate outputs based on scale, e.g. 156000000&50.2 for instance. all models sometimes change scale, e.g. table 12 where ChartGemma returned values multiplied by 10. However, Deplot is more robust to this error.
5. Occasionally, ChartGemma and Gemini 1.5 Flash swap two columns as shown in table 17. As a result, RMS score is significantly lower (f1=34%) than its fixed version (f1=83%).

In the following subsubsections, we illustrate issues related to each kind of graphs.

### 4.4.1 *Bar Charts*

1. All models are good in extracting data points of Horizontal and Vertical bar charts. However, Deplot seams better in precision and in extracting negative points as in Figure 1.

| | Country | Amount (in current US$) |
|---|---|---|
| 0 | Lao PDR | 15000000 |
| 1 | Lebanon | 270000000 |
| 2 | Lesotho | 0 |
| 3 | Liberia | 0 |
| 4 | Macedonia | 5000000 |
| 5 | Madagascar | 15000000 |

***Table 3***—Gemini 1.5 Flash prediction for PlotQA vertical bar #32640

### 4.4.2 *Line Charts*

1. Tables 13, 12 and 11 show that both models are very good in extracting data points from line charts.

[5] It is worth noting that we have not seen cases where increasing is replaced by decreasing trends or vice versa in our Test-Sample.

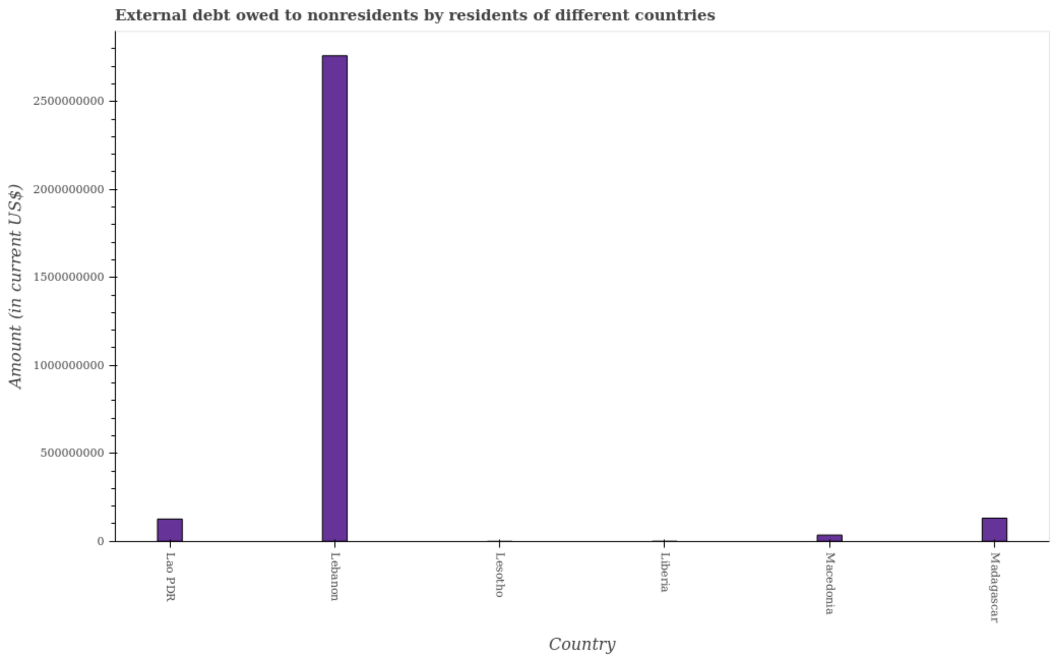

***Figure 1***—PlotQA Vertical Bar no. 32640.

| | Country | Amount (in current US$) |
|---|---|---|
| 1 | Lao PDR | 100,000,000 |
| 2 | Lebanon | 270,000,000 |
| 3 | Lesotho | 0 |
| 4 | Liberia | 0 |
| 5 | Macedonia | 30,000,000 |
| 6 | Madagascar | 150,000,000 |

***Table 4***—ChartGemma prediction for PlotQA vertical bar #32640

| Country | Debt |
|---|---:|
| Lao PDR | 123191000 |
| Lebanon | 2757587000 |
| Lesotho | -11173000 |
| Liberia | -12609000 |
| Macedonia | 33960000 |
| Madagascar | 130544000 |

***Table 5***—Deplot prediction for PlotQA vertical bar #32640

2. There are some graphs, like 3, the Gemini API just fails with no clear response message. However, it is suspected that the large number of data points might be the reason. It is noticable for the same chart to cause Deplot to repeat some rows as shown in table 9.
3. Table 16 shows that ChartGemma may fail in extracting data points from slightly complex graphs. It fails in both extracting correct values as well as mapping them to the correct label.

Tables 6 and 7 include Gemini 1.5 Flash and ChartGemma predictions for figure 2 respectively.

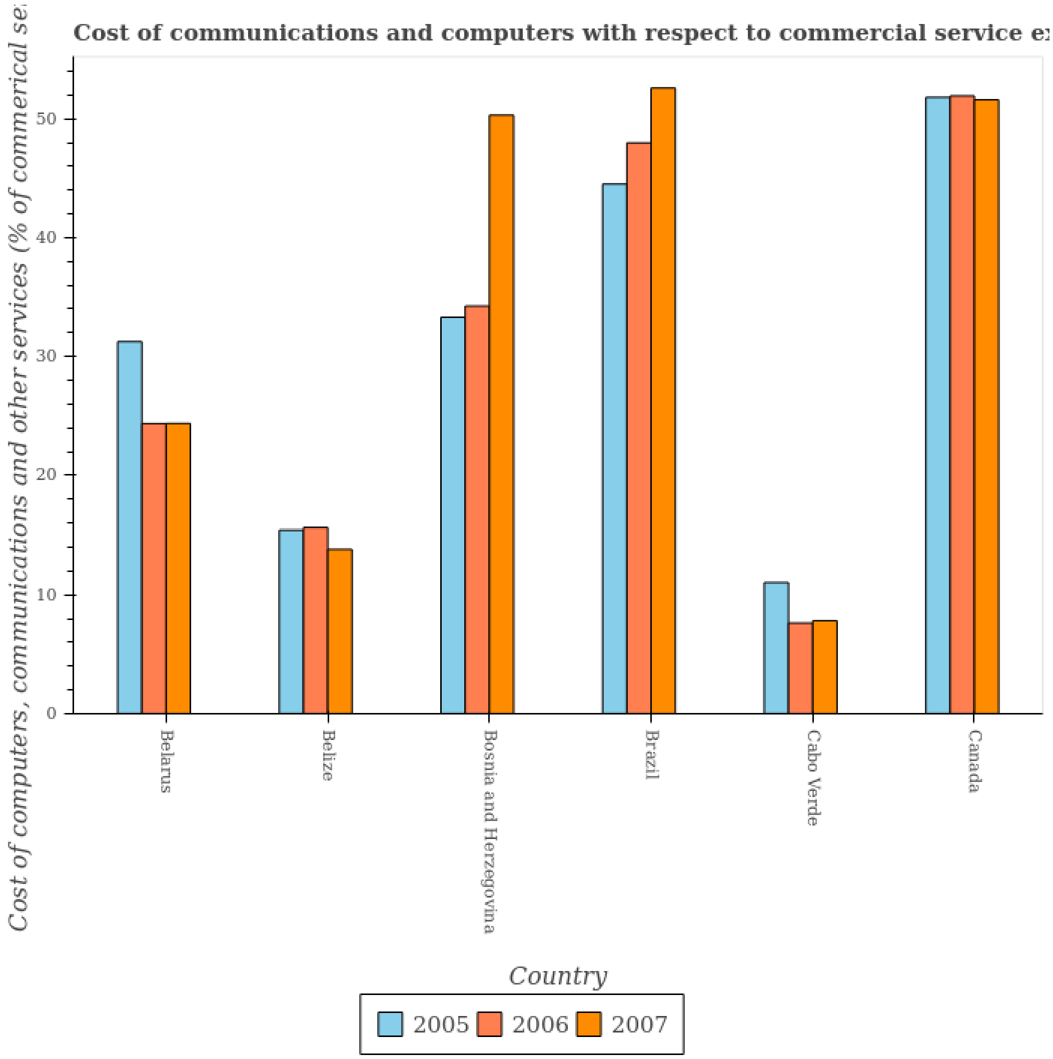

*Figure 2*—Vertical Bar Chart example from PlotQA testset.

|  | Country | 2005 | 2006 | 2007 |
|---|---|---|---|---|
| 0 | Belarus | 31 | 24 | 23 |
| 1 | Belize | 15 | 15 | 13 |
| 2 | Bosnia and Herzegovina | 33 | 34 | 52 |
| 3 | Brazil | 44 | 47 | 52 |
| 4 | Cabo Verde | 10 | 7 | 8 |
| 5 | Canada | 51 | 52 | 51 |

*Table 6*—Gemini 1.5 Flash predictions on Vertical Bar # 25905

|  | Country | 2005 Cost of computers, communications and other services (% of comm |
|---|---|---|
|  |  |  |
| 1 | Belarus | 31 |
| 2 | Belize | 15 |
| 3 | Bosnia and Herzegovina | 33 |
| 4 | Brazil | 45 |
| 5 | Cabo Verde | 11 |
| 6 | Canada | 52 |

*Table 7*—ChartGemma predictions for vertical bar image no. 2590 from PlotQA

| Country | 2005 | 2006 | 2007 |
|---|---|---|---|
| Belarus | 31.22 | 24.37 | 24.45 |
| Belize | 15.34 | 15.58 | 13.8 |
| Bosnia and Herzegovina | 33.22 | 34.26 | 50.3 |
| Brazil | 44.5 | 47.98 | 52.6 |
| Cabo Verde | 11.06 | 7.63 | 7.76 |
| Canada | 51.81 | 51.93 | 51.6 |

*Table 8*—Deplot predictions on Vertical Bar # 25905

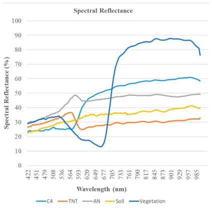

*Figure 3*—Example for charts that causes the API to fail.

| Wavelength (nm) | C4 | TNT | AN | Soil | Vegetation |
|---|---|---|---|---|---|
|  | 23.7 | 26.8 | 29.6 | 23.7 | 29.4 |
|  | 23.9 | 32.6 | 35.6 | 27.9 | 33.8 |
| 70.0 | 23.9 | 28.6 | 46.9 | 25.6 | 30.1 |
| 50.0 | 25.5 | 33.2 | 36.5 | 29.9 | 33.8 |
| 50.0 | 25.2 | 35.6 | 48.3 | 29.5 | 25.2 |
| 50.0 | 25.6 | 34.8 | 48.3 | 30.9 | 25.4 |
| 50.0 | 25.5 | 34.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |
| 50.0 | 25.8 | 33.8 | 48.3 | 30.5 | 25.2 |

***Table 9***—Deplot prediction for ICPR22 line chart #6339

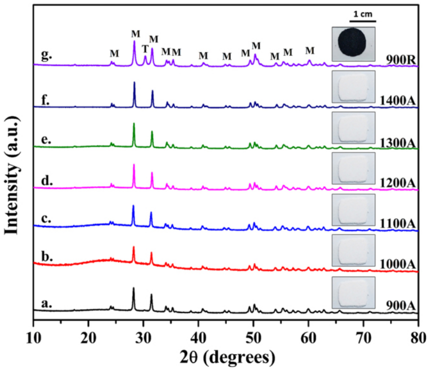

*Figure 4*—A good example for graph in the wild that causes Gemini 1.5 Flash to fail.

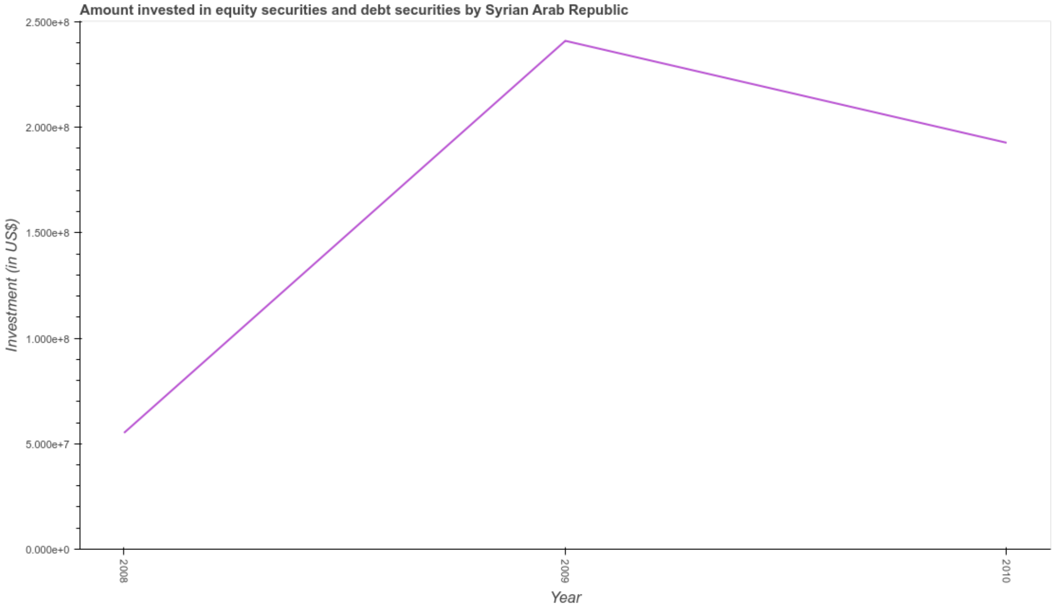

*Figure 5*—Example for Line Chart from PlotQA testset # 21673 about Portfolio Investment

| | name | color | label | bboxes |
|---|---|---|---|---|
| 0 | Portfolio Investment | #BA55D3 | Portfolio Investment | ['y': 51, 'x': 132, 'w': 466, 'h': 413, 'y': 51, ' |

*Table 10*—Reference for Line Chart from PlotQA #21673 Portfolio Investment

|   | Year | Investment (in USD) |
|---|---|---|
| 0 | 2008 | 54000000 |
| 1 | 2009 | 240000000 |
| 2 | 2010 | 200000000 |

***Table 11***—Predicted data points by Gemini 1.5 Flash for Line Chart from PlotQA #21673 Portfolio Investment

|   | Year | Investment (in USD) |
|---|---|---|
| 1 | 2008 | 500000000 |
| 2 | 2009 | 2400000000 |
| 3 | 2010 | 1900000000 |

***Table 12***—ChartGemma: prediction for PlotQA line chart #21673

| Year | Portfolio Investment |
|---|---|
| 2008 | 55000000 |
| 2009 | 241000000 |
| 2010 | 192700000 |

***Table 13***—Predicted data points by Deplot for Line Chart from PlotQA #21673 Portfolio Investment

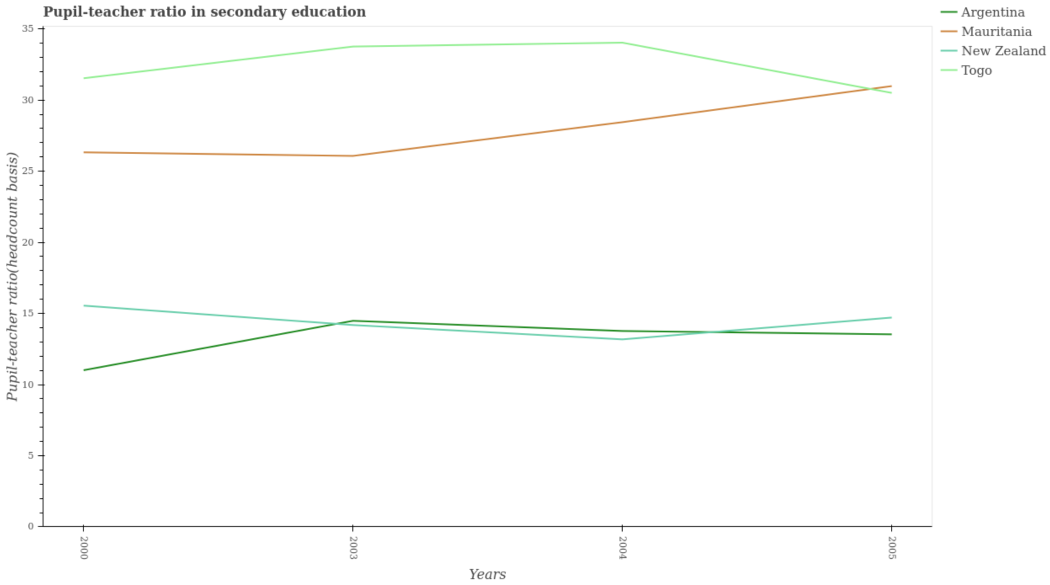

***Figure 6***—PlotQA # 20049: Line chart containing 4 lines.

|  | name | color | label | bboxes |
|---|---|---|---|---|
| 0 | Argentina | #228B22 | Argentina | ['y': 386, 'x': 101, 'w': 320, 'h': 58, 'y': 386, 'x': 421, 'w': |
| 1 | Mauritania | #CD853F | Mauritania | ['y': 186, 'x': 101, 'w': 320, 'h': 4, 'y': 150, 'x': 421, 'w': 3 |
| 2 | New Zealand | #66CDAA | New Zealand | ['y': 368, 'x': 101, 'w': 320, 'h': 23, 'y': 391, 'x': 421, 'w': |
| 3 | Togo | #90EE90 | Togo | ['y': 60, 'x': 101, 'w': 320, 'h': 38, 'y': 56, 'x': 421, 'w': 32 |

***Table 14***—Reference table for PlotQA line chart # 20049

|  | Year | Argentina | Mauritania | New Zealand | Togo |
|---|---|---|---|---|---|
| 0 | 2000 | 10.600000 | 26.000000 | 15.800000 | 31.400000 |
| 1 | 2003 | 14.200000 | 25.800000 | 14.000000 | 33.200000 |
| 2 | 2004 | 13.600000 | 28.000000 | 13.000000 | 33.800000 |
| 3 | 2005 | 13.400000 | 30.200000 | 14.600000 | 30.000000 |

***Table 15***—Gemini 1.5 Flash prediction for PlotQA line chart # 20049

|  | Years | Argentina | Mauritius | New Zealand | Togo |
|---|---|---|---|---|---|
| 1 | 2000 | 21 | 15 | 22 | 11 |
| 2 | 2003 | 21 | 14 | 23 | 14 |
| 3 | 2004 | 22 | 13 | 23 | 13 |
| 4 | 2005 | 21 | 14 | 21 | 14 |

***Table 16***—ChartGemma prediction for PlotQA line chart # 20049. The model fails in mapping lines with values, e.g. Togo column seams more likely to be Argentina. For values, it is obvious that ChartGemma is very far away from correctly detecting values greater than 20!

|   | Australia | Turkmenistan |
|---|---|---|
| 0 | 'Year': 2009.0, 'Subscribers per 100 People': 47.0 | 'Year': 2009.0, 'Subscribers per 100 People': 48.5 |
| 1 | 'Year': 2010.0, 'Subscribers per 100 People': 46.0 | 'Year': 2010.0, 'Subscribers per 100 People': 47.5 |
| 2 | 'Year': 2011.0, 'Subscribers per 100 People': 45.0 | 'Year': 2011.0, 'Subscribers per 100 People': 46.0 |
| 3 | 'Year': 2012.0, 'Subscribers per 100 People': 44.5 | 'Year': 2012.0, 'Subscribers per 100 People': 45.0 |
| 4 | 'Year': 2013.0, 'Subscribers per 100 People': 44.0 | 'Year': 2013.0, 'Subscribers per 100 People': 44.0 |

***Table 17***—Example for Gemini Flash predictions where it swapped the values of Turkmenistan and United States. The swapped table has score of F1 = 0.34 and the corrected version has F1 = 0.83.

### 4.4.3 *Scatter Plots*

1. There is no model capable of handling scatter plots correctly!

```
Ground Truth csv:
\input{test-sample/icpr22/csvs/scatter/PMC2279396___g002.csv}
Deplot output csv:
\input{test-sample/icpr22/predictions/deplot/scatter/PMC2279396___g002.csv}
Gemini 1.5 Flash output json:
\input{test-sample/icpr22/predictions/gemini-1.5-flash/1st_prompt/scatter/PMC2279396___
ChartGemma output text:
\input{test-sample/icpr22/predictions/ahmed-masry/chartgemma/initial_prompt/scatter/PMC
```

### 4.4.4 *Box Plots*

1. Gemini 1.5 Flash was the only model to correctly extract data points from vertical box plots. However, comparing Gemini 1.5 Flash's output to ground truth table needs custom logic because some graphs have more than one data series.

```
Ground Truth csv:
\input{test-sample/icpr22/csvs/vertical-box/PMC1855992___g001.csv}
Deplot prediction:
\input{test-sample/icpr22/predictions/deplot/vertical-box/PMC1855992___g
Gemini 1.5 Flash prediction:
\input{cat  test-sample/icpr22/predictions/gemini-1.5-flash/1st_prompt/v
```

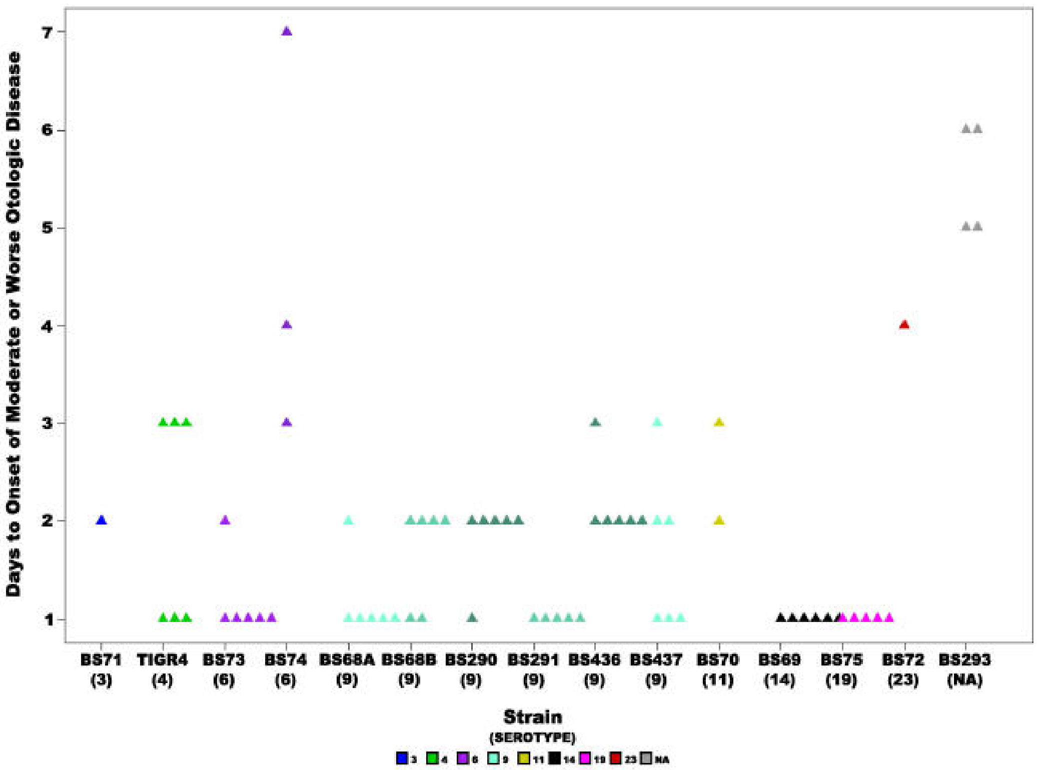

*Figure 7*—Scatter Plot from ICPR22 testset #2279

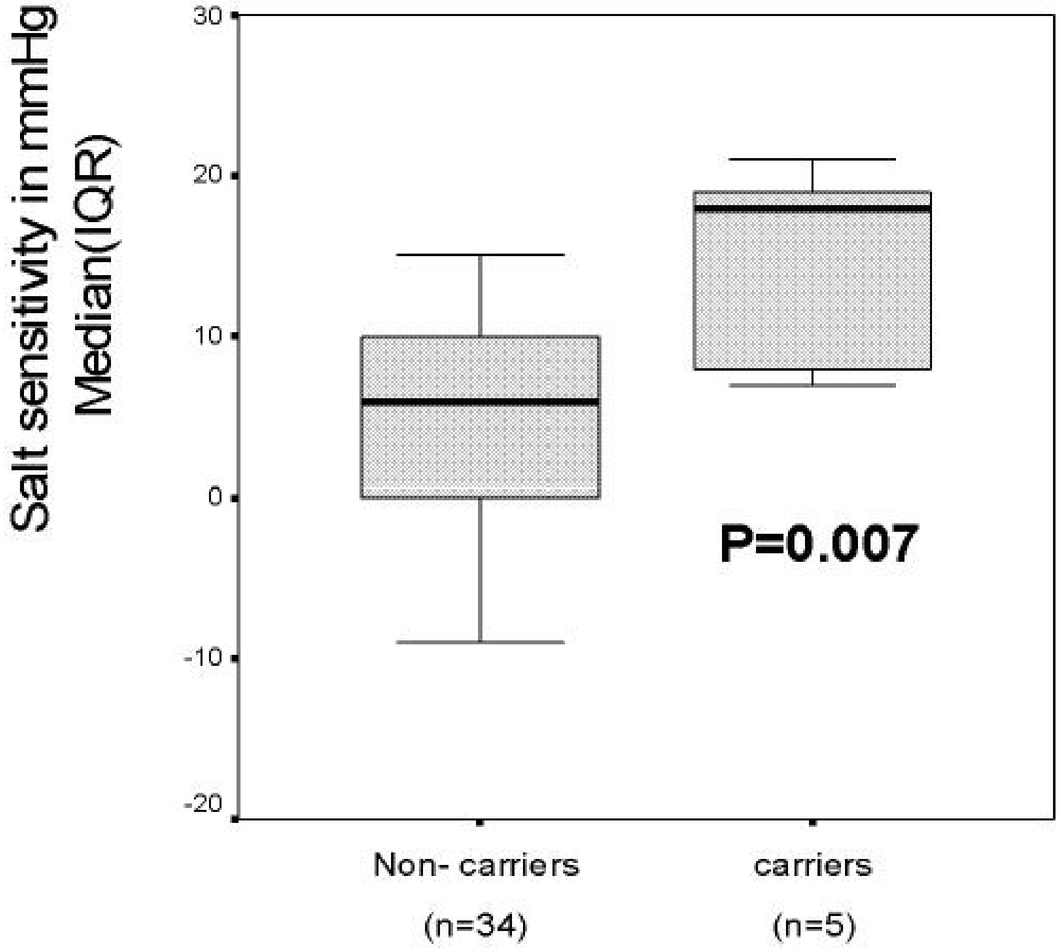

*Figure 8*—Vertical Box Plot from ICPR22 testset #1855

```
ChartGemma’s  output markdown:
\input{test-sample/icpr22/predictions/ahmed-masry/chartgemma/initial_pro
```

## 5 CONCLUSION AND RECOMMENDATIONS

In conclusion, our study provides a comprehensive evaluation of the performance of various models on chart-to-table Conversion. Our results show that while some models excel in certain types of charts, none of them perform evenly across all chart types. This highlights the complexity and diversity of chart-to-table tasks, which require specialized models or very capable large general models.

Our findings have important implications for researchers and practitioners working on data extraction and analysis. They suggest that future research should focus on developing more robust and flexible models that can handle a wide range of chart types and styles. Additionally, our results emphasize the need for larger and more diverse datasets to train and evaluate these models.

Overall, our study contributes to a deeper understanding of the challenges and opportunities in chart-to-table tasks and provides a foundation for future research in this area. We hope that our findings will fosters further innovation and improvement in data extraction and analysis.

## 6 LIMITATIONS

One limitation of our analysis is testing pre-trained models exclusively and did not test models like ChartOCR. All those models are not directly comparable and have no existing weights and implementation to easily test. This has been left for future work.

## 7 ACKNOWLEDGMENTS

This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) PACE, 2017 at the Georgia Institute of Technology, Atlanta, Georgia, USA.

## 8 REFERENCES

1. Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems* 28.
2. Jung, Daekyoung, Kim, Wonjae, Song, Hyunjoo, Hwang, Jeong-in, Lee, Bongshin, Kim, Bohyoung, and Seo, Jinwook (2017). "Chartsense: Interactive data extraction from chart images". In: *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 6706–6717.
3. PACE, M (2017). "Partnership for an advanced computing environment (PACE)". In.
4. Poco, Jorge and Heer, Jeffrey (2017). "Reverse-engineering visualizations: Recovering visual encodings from chart images". In: *Computer graphics forum*. Vol. 36. 3. Wiley Online Library, pp. 353–363.
5. Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, and Polosukhin, Illia (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.
6. Herzig, Jonathan, Nowak, Paweł Krzysztof, Müller, Thomas, Piccinno, Francesco, and Eisenschlos, Julian Martin (2020). "TaPas: Weakly supervised table parsing via pre-training". In: *arXiv preprint arXiv:2004.02349*.
7. Methani, Nitesh, Ganguly, Pritha, Khapra, Mitesh M, and Kumar, Pratyush (2020). "Plotqa: Reasoning over scientific plots". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1527–1536.
8. Raffel, Colin, Shazeer, Noam, Roberts, Adam, Lee, Katherine, Narang, Sharan, Matena, Michael, Zhou, Yanqi, Li, Wei, and Liu, Peter J (2020). "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140, pp. 1–67.
9. Luo, Junyu, Li, Zekun, Wang, Jinpeng, and Lin, Chin-Yew (2021). "Chartocr: Data extraction from charts images via a deep hybrid framework". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 1917–1925.
10. Sreevalsan-Nair, Jaya, Dadhich, Komal, and Daggubati, Siri Chandana (2021). "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*. Springer, pp. 219–241.

11. Kumar, Anukriti, Ganu, Tanuja, and Guha, Saikat (2022). "ChartParser: Automatic Chart Parsing for Print-Impaired". In: *arXiv preprint arXiv:2211.08863*.
12. Li, Shufan, Lu, Congxi, Li, Linkai, and Zhou, Haoshuai (2022). "Chart-RCNN: Efficient Line Chart Data Extraction from Camera Images". In: *arXiv preprint arXiv:2211.14362*.
13. Liu, Fangyu, Eisenschlos, Julian Martin, Piccinno, Francesco, Krichene, Syrine, Pang, Chenxi, Lee, Kenton, Joshi, Mandar, Chen, Wenhu, Collier, Nigel, and Altun, Yasemin (2022a). "Deplot: One-shot visual language reasoning by plot-to-table translation". In: *arXiv preprint arXiv:2212.10505*.
14. Liu, Fangyu, Piccinno, Francesco, Krichene, Syrine, Pang, Chenxi, Lee, Kenton, Joshi, Mandar, Altun, Yasemin, Collier, Nigel, and Eisenschlos, Julian Martin (2022b). "Matcha: Enhancing visual language pretraining with math reasoning and chart derendering". In: *arXiv preprint arXiv:2212.09662*.
15. Masry, Ahmed, Long, Do Xuan, Tan, Jia Qing, Joty, Shafiq, and Hoque, Enamul (2022). "Chartqa: A benchmark for question answering about charts with visual and logical reasoning". In: *arXiv preprint arXiv:2203.10244*.
16. Cheng, Zhi-Qi, Dai, Qi, and Hauptmann, Alexander G (2023). "Chartreader: A unified framework for chart derendering and comprehension without heuristic rules". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22202–22213.
17. Lee, Kenton, Joshi, Mandar, Turc, Iulia Raluca, Hu, Hexiang, Liu, Fangyu, Eisenschlos, Julian Martin, Khandelwal, Urvashi, Shaw, Peter, Chang, Ming-Wei, and Toutanova, Kristina (2023). "Pix2struct: Screenshot parsing as pretraining for visual language understanding". In: *International Conference on Machine Learning*. PMLR, pp. 18893–18912.
18. Masry, Ahmed, Kavehzadeh, Parsa, Do, Xuan Long, Hoque, Enamul, and Joty, Shafiq (2023). "Unichart: A universal vision-language pretrained model for chart comprehension and reasoning". In: *arXiv preprint arXiv:2305.14761*.
19. Rousseau, Jean-Jacques and Kapralos, Bill (2023). *Pattern Recognition, Computer Vision, and Image Processing. ICPR 2022 International Workshops and Challenges: Montreal, QC, Canada, August 21–25, 2022, Proceedings, Part I*. Vol. 13643. Springer Nature.
20. Touvron, Hugo, Martin, Louis, Stone, Kevin, Albert, Peter, Almahairi, Amjad, Babaei, Yasmine, Bashlykov, Nikolay, Batra, Soumya, Bhargava, Prajjwal, Bhosale, Shruti, et al. (2023). "Llama 2: Open foundation and fine-tuned chat models". In: *arXiv preprint arXiv:2307.09288*.

21. Voigt, Henrik, Carvalhais, Nuno, Meuschke, Monique, Reichstein, Markus, Zarrie, Sina, and Lawonn, Kai (2023). "VIST5: An adaptive, retrieval-augmented language model for visualization-oriented dialog". In: *The 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 70–81.

22. Beyer, Lucas, Steiner, Andreas, Pinto, André Susano, Kolesnikov, Alexander, Wang, Xiao, Salz, Daniel, Neumann, Maxim, Alabdulmohsin, Ibrahim, Tschannen, Michael, Bugliarello, Emanuele, et al. (2024). "Paligemma: A versatile 3b vlm for transfer". In: *arXiv preprint arXiv:2407.07726*.

23. Cao, Yukun, Han, Shuo, Gao, Zengyi, Ding, Zezhong, Xie, Xike, and Zhou, S Kevin (2024). "Graphinsight: Unlocking insights in large language models for graph structure understanding". In: *arXiv preprint arXiv:2409.03258*.

24. Chung, Hyung Won, Hou, Le, Longpre, Shayne, Zoph, Barret, Tay, Yi, Fedus, William, Li, Yunxuan, Wang, Xuezhi, Dehghani, Mostafa, Brahma, Siddhartha, et al. (2024). "Scaling instruction-finetuned language models". In: *Journal of Machine Learning Research* 25.70, pp. 1–53.

25. Masry, Ahmed, Shahmohammadi, Mehrad, Parvez, Md Rizwan, Hoque, Enamul, and Joty, Shafiq (2024a). "ChartInstruct: Instruction Tuning for Chart Comprehension and Reasoning". In: *arXiv preprint arXiv:2403.09028*.

26. Masry, Ahmed, Thakkar, Megh, Bajaj, Aayush, Kartha, Aaryaman, Hoque, Enamul, and Joty, Shafiq (2024b). "ChartGemma: Visual Instruction-tuning for Chart Reasoning in the Wild". In: *arXiv preprint arXiv:2407.04172*.

27. Team, Gemini, Georgiev, Petko, Lei, Ving Ian, Burnell, Ryan, Bai, Libin, Gulati, Anmol, Tanzer, Garrett, Vincent, Damien, Pan, Zhufeng, Wang, Shibo, et al. (2024). "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context". In: *arXiv preprint arXiv:2403.05530*.

28. Goswami, Kanika, Mathur, Puneet, Rossi, Ryan, and Dernoncourt, Franck (2025). "ChartCitor: Multi-Agent Framework for Fine-Grained Chart Visual Attribution". In: *arXiv preprint arXiv:2502.00989*.

29. Zhao, Xuanle, Luo, Xianzhen, Shi, Qi, Chen, Chi, Wang, Shuo, Che, Wanxiang, Liu, Zhiyuan, and Sun, Maosong (2025). "ChartCoder: Advancing Multimodal Large Language Model for Chart-to-Code Generation". In: *arXiv preprint arXiv:2501.06598*.

# A EVALUATION METRICS

## A.1 Relative Mapping Similarity (RMS)

```
Require: $prp$ (Predicted Table), $t$ (Ground Truth Table)
Ensure: $rms\_precision, rms\_recall, rms\_f1$
    Compute Normalized Edit Distance between $p_r||p_c$ and $t_r||t_c$ , where || is concatenation operator. Compute pairwise similarities matrix make binarized similarities matrix by inserting 1 in place of highest similarities and zeros otherwise.
for do$p_i$ in p.values:
    for do$t_j$ in t.values: $d_{theta_{ij}} = min(1, \frac{|p_i - t_j|}{|t_j|})$ $d_{tao_{theta_{ij}}} = (1 - NormalizedEditDistance) * (1 - d_{theta})$
    end for
end for$RMS_{precision} = \sum_i \sum_j d_{tao_{theta_{ij}}}/len(p)$ $RMS_{recall} = \sum_i \sum_j d_{tao_{theta_{ij}}}/len(t)$ $RMS_{f1} = 2 * \frac{RMS_{precision} * RMS_{recall}}{RMS_{precision} + RMS_{recall}}$
```

## A.2 metrics introduced in ChartOCR

In this subsection, we illustrate metrics introduced by Luo et al., 2021.

### A.2.1 *Bar Chart Metric*

Given a predicted bounding box $p = [x_p, y_p, w_p, h_p]$ and a ground truth box $g = [x_g, y_g, w_g, h_g]$, the distance function $D(p, g)$ is defined as:

$$D(p, g) = \min\left(1, \left|\frac{x_p - x_g}{w_g}\right| + \left|\frac{y_p - y_g}{h_g}\right| + \left|\frac{h_p - h_g}{h_g}\right|\right)$$

Width $w$ is ignored because it does not affect chart reading.

Construct a pairwise cost matrix $C$ between predicted and ground truth boxes, then solve a minimum cost assignment problem using the Hungarian algorithm. The final score is:

$$\text{Score} = 1 - \frac{\text{min cost}}{K}, \quad \text{where } K = \max(N, M)$$

Here $N$ and $M$ are the number of predicted and ground truth boxes respectively.

### A.2.2 *Line Chart Metric*

Let $P = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ be the predicted points and $G = \{(u_1, v_1), \ldots, (u_M, v_M)\}$ be the ground truth. The recall is defined as:

$$\mathrm{Rec}(P, G) = \frac{1}{u_M - u_1} \sum_{i=1}^{M} (1 - \mathrm{Err}(v_i, u_i, P)) \cdot \mathrm{Intv}(i, G)$$

Where:

$$\mathrm{Err}(v_i, u_i, P) = \min\left(1, \left|\frac{v_i - I(P, u_i)}{v_i}\right|\right)$$

$I(P, u_i)$ is the interpolated value of $P$ at $x = u_i$. $\mathrm{Intv}(i, G)$ is the interval weight:

$$\mathrm{Intv}(i, G) = \begin{cases} \frac{u_{i+1} - u_i}{2} & \text{if } i = 1 \\ \frac{u_i - u_{i-1}}{2} & \text{if } i = M \\ \frac{u_{i+1} - u_{i-1}}{2} & \text{otherwise} \end{cases}$$

Precision is defined similarly. The final score is the F1-score:

$$F1 = \frac{2 \cdot \mathrm{Prec} \cdot \mathrm{Rec}}{\mathrm{Prec} + \mathrm{Rec}}$$

### A.2.3 *Pie Chart Metric*

Let $P = [x_1, \ldots, x_N]$ and $G = [y_1, \ldots, y_M]$ be the predicted and ground-truth value sequences in clockwise order. The match score is computed via dynamic programming:

$$\mathrm{score}(i, j) = \max\left(\mathrm{score}(i{-}1, j), \mathrm{score}(i, j{-}1), \mathrm{score}(i{-}1, j{-}1) + 1 - \left|\frac{x_i - y_j}{y_j}\right|\right)$$

with $\mathrm{score}(0, j) = \mathrm{score}(i, 0) = 0$ for all $i, j$. The final normalized matching score is:

$$\mathrm{Score} = \frac{\mathrm{score}(N, M)}{M}$$