ChartReader: A Unified Framework for Chart Derendering and Comprehension without Heuristic Rules

Zhi-Qi Cheng\textsuperscript{1} Qi Dai\textsuperscript{2}\,\, Alexander G. Hauptmann\textsuperscript{1}
\textsuperscript{1}Language Technologies Institute, Carnegie Mellon University  \textsuperscript{2}Microsoft Research

zhiqic@cs.cmu.edu, qid@microsoft.com, alex@cs.cmu.edu

Abstract

Charts are a powerful tool for visually conveying complex data, but their comprehension poses a challenge due to the diverse chart types and intricate components. Existing chart comprehension methods suffer from either heuristic rules or an over-reliance on OCR systems, resulting in suboptimal performance. To address these issues, we present ChartReader, a unified framework that seamlessly integrates chart derendering and comprehension tasks. Our approach includes a transformer-based chart component detection module and an extended pre-trained vision-language model for chart-to-X tasks. By learning the rules of charts automatically from annotated datasets, our approach eliminates the need for manual rule-making, reducing effort and enhancing accuracy. We also introduce a data variable replacement technique and extend the input and position embeddings of the pre-trained model for cross-task training. We evaluate ChartReader on Chart-to-Table, ChartQA, and Chart-to-Text tasks, demonstrating its superiority over existing methods. Our proposed framework can significantly reduce the manual effort involved in chart analysis, providing a step towards a universal chart understanding model. Moreover, our approach offers opportunities for plug-and-play integration with mainstream LLMs such as T5 and TaPus, extending their capability to chart comprehension tasks.\textsuperscript{1}

1. Introduction

The adage, “a picture is worth a thousand words,” underscores the immense value of charts found on various websites and articles, which often depict knowledge that cannot be conveyed through words alone. Chart derendering, which refers to the conversion of charts into tables (i.e., Chart-to-Table \cite{11, 41, 42}), is widely viewed as essential in facilitating a range of downstream tasks, such as chart question-answering (ChartQA \cite{27, 45, 66}) and chart summarization (Chart-to-Text \cite{7, 15, 48}). As shown in Figure 1, the Chart-to-Table task aims to recognize the chart as a machine-readable table, while ChartQA and Chart-to-Text tasks involve answering pre-set questions and summarizing the content of the chart, respectively. [Best viewed in color].

Figure 1. Illustration of chart derendering and comprehension tasks. The Chart-to-Table task aims to transform a chart into a machine-readable table, while ChartQA and Chart-to-Text tasks involve answering questions and summarizing the content of the chart, respectively. [Best viewed in color].

\textsuperscript{1}Code is at https://github.com/zhiqic/ChartReader

\footnotesize{\textsuperscript{1}Corresponding author}
Figure 2. Demonstrates the complexity of charts in EC400K [42], which can vary in type, design, and visual properties. Charts can contain intricate components, texture variations, and speckled backgrounds, posing challenges for chart comprehension. [Best viewed in color].

bar [13, 50] and line [30, 43] charts. These limitations impede the ability to extract data for unknown categories. As a result, many latest methods [40, 45] even use tables directly from ground truth to complete the answers and summary tasks. It is evident that these studies are challenging to automate and struggle to extract data from real-world charts.

On the other hand, current chart comprehension methods, such as Chart-to-Text [7, 15, 48] and ChartQA [29, 36, 44, 45, 54], often heavily rely on off-the-shelf OCR systems or pre-extracted tables from the ground truth. By treating chart derendering as a black box, these methods neglect the visual and structural information of the charts, resulting in the following issues: (1) Chart-to-Text and ChartQA tasks devolve into text-only quizzes [39, 44], as they cannot extract visual semantics from chart derendering. This explains why OCR-based and end-to-end methods, such as LayoutLM [64], PresSTU [31], PaLI [8], CoCa [65], Donut [32], and Dessurt [14], have shown suboptimal results in chart understanding. (2) Chart-to-table tasks do not benefit from chart comprehension tasks. Due to the lack of understanding to the visual semantics in charts, existing systems, such as those using OCR systems [39, 42], struggle to accurately convert charts to tables. Overall, we contend that the problems with chart comprehension arise from an over-reliance on predefined rules and a lack of a universal framework to support multi-tasking.

In light of the previous analysis, it seems that a visual-language model is a promising direction for building a universal framework. However, while Pix2Struct [35], a pre-training strategy for visually-situated language, has shown superior performance over OCR-based models [2, 5, 37, 60], it is not suitable for chart derendering. Moreover, despite recognizing this issue, Metcha [40] had to rely on Pix2Struct as a backbone due to the lack of better visual-language models. Nonetheless, neither Pix2Struct nor Metcha address the two issues identified earlier: 1) excessive reliance on heuristic rules in table derendering, and 2) heavy reliance on existing OCR systems despite attempts to incorporate additional chart comprehension tasks.

To overcome these concerns, we introduce ChartReader, a unified framework that seamlessly integrates chart derendering and comprehension tasks. Our approach comprises a rule-free chart component detection module and an extended pre-trained vision-language model for chart-to-X (text/table/QA) tasks. Unlike heuristic rule-based methods, our approach leverages a transformer-based approach to detect the location and type of chart components, enabling automatic rule learning by processing existing annotated datasets. To enhance cross-task training, we extend the input and position embeddings of the pre-trained model and introduce a data variable replacement technique. Specifically, we standardize chart-to-X (table/text) tasks as question-answering problems, allowing us to solve multiple chart understanding tasks effectively. Additionally, the model generates the data variable instead of the actual value to avoid errors and hallucinations, which improves the consistency in multi-task training. Our approach represents a step towards a unified chart understanding model, as validated through experiments. The proposed framework has the potential to reduce the manual effort involved in chart analysis, paving the way for more efficient and accurate chart comprehension.

To summarize, our contributions are: 1) a unified framework that seamlessly integrates chart derendering and comprehension tasks; 2) a rule-free chart component detection module that leverages a transformer-based approach to automatically learn the rules of charts; 3) extending the input and position embeddings of the pre-trained LLMs and employing a data variable replacement technique to improve cross-task training; 4) validating our approach through experiments, demonstrating significant improvement over existing methods in chart understanding tasks.

2. Related Works

This section reviews related works on chart understanding, including chart derendering, question-answering, and summarization, highlighting the differences from previous work and the impact on visual-language research.

Chart Derendering, also known as Chart-to-Table, involves identifying the constituent components of an image of a chart, such as bars, pies, and legends, to extract the underlying data represented by the chart. Traditional methods [4, 18, 25, 49, 52, 53] relied on hand-designed rules based on edges and colors, which were time-consuming and not easily generalized to new chart types. Deep-learning based approaches [11, 41, 42] utilizing object detection and
Chart Question-Answering, or ChartQA, is the task of LLMs to eliminate heuristic rules and tackle various chart types. In contrast, our approach is an end-to-end framework that integrates component detection and utilizes fine-tuned LLMs to eliminate heuristic rules and tackle various chart comprehending tasks. 

3. ChartReader Framework

Our unified framework aims to support various chart analysis tasks, including chart-to-table, chart-to-text, and chartQA. As shown in Figure 3, it consists of two main components: (1) a rule-free chart component detection module, and (2) an extended pre-trained vision-language model for chart-to-X (text/table/QA) tasks. We will delve into the functions of each module in detail in the upcoming sections.

3.1. Chart Component Detection

We exploit a transformer-based approach to detect the location and type of chart components without relying on heuristic rules. Our approach consists of three main steps: 1) center/keypoint detection, 2) center/keypoint grouping, and 3) component position/type prediction.

Overcoming Heuristics. The motivation behind this is to overcome the limitations of heuristic rule-based methods in handling various chart styles that heavily rely on domain-specific knowledge and complex rule definitions. By processing annotated existing datasets, our model can automatically learn the rules of charts. Furthermore, the optimized framework can seamlessly be applied to other downstream chart understanding tasks. As illustrated in Figure 4, our updated model relies on center and key points inferred from existing datasets instead of heuristic rules to locate chart components and overcome the effects of style variations. We convert the upper left and lower right corners and position centers of each component into the center and key points, respectively. Although our approach still requires a large amount of annotated data, we do not need to design specific rules for each chart type. Our approach represents a step towards a unified chart understanding model and has been validated through experiments.

Step-1: Center/Keypoint Detection. We detect the centers $p_c \in \mathcal{C}$ and keypoints $p_k \in \mathcal{K}$ of chart components us-
mizing the center point position using the grouping score
dict the position and type of each chart component by opti-
chart components.

Figure 3. illustrates the components of our proposed unified framework for various chart analysis tasks, including chart-to-table, chart-
to-text, and chartQA. The framework is comprised of two main modules: (1) a transformer-based chart component detection module that eliminates the need for manual rule-making, and (2) an extended pre-trained vision-language model that enables chart-to-X (text/table/QA) tasks. The combination of these two modules enables seamless integration and efficient execution of chart comprehension tasks.

**Step-2: Center/Keypoint Grouping.** For each detected center and keypoint, we extract the features of their corresponding positions to obtain initial embeddings. To better determine the chart component, we introduce a type toke
φ\textsubscript{p}\text{and} φ\textsubscript{k} and use multi-head attention to obtain the weights of the center point \textit{p}\textsubscript{c} and keypoint \textit{p}\textsubscript{k} as,

\[
G(p_c, p_k) = (W_C[h_{p_c}, \phi_{p_c}])^T W_K[h_{p_k}, \phi_{p_k}],
\]

where \(h_{p_c}\) and \(h_{p_k}\) respectively represent the hidden features of the center point and keypoint. The matrices \(W_C\) and \(W_K\) are projection matrices of the hidden features of the center point and the key point. We use fixed sinusoidal features to encode the absolute x- and y-axis positions and add \(\phi_{p_c}\) and \(\phi_{p_k}\) to the embedding before multiplication. After obtaining the weights of \(G\) using Eqn. 1, we compute the final grouping score by normalizing the weights with a softmax function over the entire set of keypoints as,

\[
\text{attn}(p_c, p_k) = \frac{\exp(G(p_c, p_k))}{\sum_{k \in K} \exp(G(p_c, p_k))},
\]

where the softmax function sorts keypoints \textit{k} \in \textit{K} from the same component and filters the most relevant ones (i.e., \textit{p} \textsubscript{k}) for each center point \textit{p}\textsubscript{c}. This approach enables effective grouping of centers and keypoints to their corresponding chart components.

**Step-3: Component Position & Type Prediction.** We predict the position and type of each chart component by optimizing the center point position using the grouping score and keypoint positions from the Hourglass network [47] without the corner pooling layer to improve generalization across chart styles. The location \text{loc}_{\textit{p}_\text{c}} \in \mathbb{R}^2 and component type \textit{y}_\text{c} \in \{1, ..., T\} for all centers and keypoints are predicted using the focal loss \(L_{\text{focal}}\) [34] and the smooth \(L_1\) loss [19]. We use the central pooling layer for all component types, rather than different pooling strategies for each component type [42], to effectively detect center and keypoint of chart components in various styles.

This supervises the optimization process and helps to accurately predict the position of each chart component. To predict the type of chart component, we compute a weighted sum of the keypoint embeddings using the grouping score as,

\[
\tilde{h}_{p_c} = \sum_{k \in K} \text{attn}(p_c, p_k) W_K[h_{p_k}, \phi_{p_k}].
\]

The resulting center point embedding \(\tilde{h}_{p_c}\) is then passed through an MLP layer to obtain the predicted probability distribution over all component types, which is then compared with the ground truth component type labels \textit{y}_\text{c} using the cross-entropy loss \(L_{\text{CLS}}\) as,

\[
L_{\text{CLS}} = -\sum_{c \in C} y_c \log(\text{softmax}(\tilde{h}_{p_c})).
\]

This supervises the optimization process and helps to accurately predict the type of each chart component.

3.2. Chart Derendering and Comprehension

We propose a unified chart understanding framework that handles Chart-to-X (text/table/QA) tasks as chartQA tasks, as illustrated in Figure X. To improve cross-task training, we extend (1) the input and position embeddings of the
where $k$ denotes layer normalization [3]. Firstly, the positional embedding $z_k$ is defined as,

$$z_k = \text{LN}(k + \text{typ})$$

where $\text{LN}()$ denotes layer normalization [3]. Secondly, the input embedding includes the type $z_k^{\text{typ}}$, location $z_k^{\text{loc}}$, and appearance $z_k^{\text{app}}$ of chart components, as well as the token $z_k^{\text{token}}$ obtained from other textual information in the dataset.

Specifically, the $z_k^{\text{token}}$ is generated by tokenizing the concatenation of textual words and is denoted as,

$$x^{\text{token}} = \{(S), [w_1], ..., [w_m], [SEP], [y_1^{\text{loc}}, w_1, 1], ..., [y_N^{\text{loc}}, w_N, 1], ..., [y_r^{\text{loc}}, w_r, 1], \}$$

where $\{S\}$ is used to distinguish between questions and answers, and $[\text{SEP}]$ indicates the presence of chart context. To incorporate information about chart components, a special token $y_r^{\text{typ}}$ is introduced to represent the chart component type $y_r^{\text{typ}} \in \{1, ..., T\}$ (such as $\text{[Axes]}$; see Figure 4). Other textual tokens obtained from each chart component are denoted as $\{w_1, ..., w_m\}$.

To capture semantic information about the chart, we incorporate a learnable one-hot embedding $z_k^{\text{typ}}$ for chart component type. The location of the $k$-th token within the chart is denoted by $x_k^{\text{loc}}$, which is a 4-dimensional feature based on the relative bounding box coordinates as,

$$x_k^{\text{loc}} = (x_{k, \text{min}} / W_{\text{im}}, y_{k, \text{min}} / H_{\text{im}}, x_{k, \text{max}} / W_{\text{im}}, y_{k, \text{max}} / H_{\text{im}})$$

where $(x_{k, \text{min}}, y_{k, \text{min}})$ and $(x_{k, \text{max}}, y_{k, \text{max}})$ represent the coordinates of the top-left and bottom-right corners of the bounding box of the $k$-th token, while $W_{\text{im}}$ and $H_{\text{im}}$ represent the width and height of the chart, respectively. To take into account the appearance of each chart component, we concatenate the features of the center and corresponding keypoints to obtain the final appearance embedding $z_k^{\text{app}}$.

Data Variable Replacement Technique. We enhance the training process by incorporating the data variable substitution technique. Its pseudo-code is presented in Alg. 1. During training, the model generates the corresponding data variable instead of the actual value, avoiding errors and hallucinations that can result from treating data records as regular tokens. For example, numerical values in the table cells are replaced with data variables such as “product1,” “product2,” etc. This approach improves the accuracy and factual consistency of generated summaries, tables, and answers, particularly when multiple data records are involved.

To supervise the use of data variables, we introduce a new loss term that penalizes the model for generating tokens that do not match any data variable. The loss term uses a function $D(x)$ that returns the data variable that matches the token $x$ if any and null otherwise. The loss term is defined as follows,

$$L_{\text{var}} = -\frac{1}{T} \sum_{t} \sum_{i \in V} \sum_{j \in E} I_{D(x_i) = v_j} \log P_{i,j}(t)$$

Figure 4. demonstrates the conversion of chart components into center and key points, which are inferred from existing datasets and used to locate chart components in our updated model. This approach eliminates the need for heuristic rules and allows the model to overcome the effects of style variations. Specifically, we convert the upper left and lower right corners of each component into center and key points, respectively. [Best viewed in color].

Pre-trained model and (2) employ a data variable replacement technique.

Motivation & Reasoning. Our approach is motivated by two main reasons: Firstly, we treat both the chart-to-X (table/text) tasks as question-answering problems. Specifically, for chart-to-table, the combination of axis labels and legends forms the question, and the extracted components serve as the answer. Similarly, chart-to-text is treated as a Q&A task to fill in blank templates generated by removing redundant information. By standardizing these previously independent tasks as Q&A tasks, we can effectively train and solve multiple chart understanding tasks. Secondly, recent advancements in these tasks have adopted sequence-to-sequence models. To address all of them in one framework, we extend pre-trained LLMs to diverse chart comprehension tasks. However, integrating these tasks is challenging, and thus, we conduct numerous experiments to determine how to extend input and position encoding. Additionally, we propose a data variable replacement technique to enhance the consistency of multi-task training. Our findings provide possibilities for extending pre-trained LLMs to chart comprehension, and it can be seamlessly extended to mainstream LLMs such as T5 and TaPas.

Positional & Input Embedding. To support tasks such as Chart-to-Table, Chart-to-Text, and ChartQA, we have extended the input and positional embeddings from natural language to chart-style data. The fused embedding $z_k$ at the $k$-th position of the sequence is defined as,

$$z_k = \text{LN}(z_k^{\text{pos}} + (z_k^{\text{token}} + z_k^{\text{typ}} + z_k^{\text{loc}} + z_k^{\text{app}}))$$

where $\text{LN}()$ denotes layer normalization [3]. Firstly, the positional embedding $z_k^{\text{pos}}$ is set to zero, as modern vision-language pre-trained models use relative position embeddings. Secondly, the input embedding includes the type $z_k^{\text{typ}}$, location $z_k^{\text{loc}}$, and appearance $z_k^{\text{app}}$ of chart components, as well as the token $z_k^{\text{token}}$ obtained from other textual information in the dataset.
Algorithm 1 Data Variable Replacement Technique
1: Input: Data $D$, Labels $Y$, Model $M$
2: Output: Trained model $M$
3: Initialize model parameters $\theta$
4: Choose hyperparameter $\alpha$
5: for each epoch do
6: for each $x, y$ in $D, Y$ do
7: Replace numerical values in $x$ with data variables to create $x'$
8: Get model prediction $M(x')$
9: Calculate $L_{\text{ans}}$ based on $M(x')$ and $y$
10: Calculate $L_{\text{var}}$ using Equation (8)
11: Total loss $L = L_{\text{ans}} + \alpha L_{\text{var}}$
12: Update $\theta$ using gradient descent on $L$
13: end for
14: end for
15: procedure INFEERENCE($x$)
16: Replace numerical values in $x$ with data variables to create $x'$
17: Get model prediction $M(x')$
18: Replace data variables in $M(x')$ with original values
19: return $M(x')$
20: end procedure

where $T$ is the length of the generated output, $N_t$ is the number of tokens in the $t$-th output, and $P_{x,j}(t)$ is the probability of generating the $i$-th token as the $j$-th data variable at the $t$-th time step. $I_{D(x_i)=v_j}$ is an indicator function that equals 1 if $D(x_i) = v_j$ and 0 otherwise.

The final optimization is to optimize the sum of the two losses, with $\alpha$ as a hyperparameter that balances the two losses,

$$L = L_{\text{ans}} + \alpha L_{\text{var}},$$

where $\alpha$ is a hyperparameter that balances the two losses. The loss $L_{\text{ans}}$ may be slightly adjusted depending on the specific chart understanding task. In summary, our approach using data variable substitution can simultaneously support chart-to-text, chart-to-table, and chartQA tasks. The introduced loss term also ensures the correct use of data variables during training.

4. Experiments

We conducted experiments on multiple chart understanding tasks, including Chart-to-Table, ChartQA, and Chart-to-Text, as shown in Table 1.

### 4.1. Evaluation Tasks and Datasets

**Chart-to-Table Task.** To evaluate the effectiveness of our approach in the Chart-to-Table task, we used the EC400K dataset [42]. This dataset contains 386,966 real-world chart images from public Excel sheets and provides both bounding box locations and numerical readings of the charts. The EC400K dataset offers a wide variety of chart types and styles, surpassing previous datasets used in chart comprehension research. This enables us to validate the performance on diverse and challenging real-world chart data.

**ChartQA Task.** We evaluated our approach for ChartQA on four datasets: FQA [28], DVQA [26], PlotQA [45], and ChartQA [44]. The FQA dataset includes charts with templates for binary answer questions and has two validation sets and two non-publicly available test sets. DVQA is a synthetic dataset that provides precise location and appearance of visual elements and metadata, including two test tasks: Test-Familiar and Test-Novel. PlotQA is a large and publicly available dataset for chart comprehension tasks, containing charts generated from real-world data. It provides two benchmarks, PlotQA-V1 and PlotQA-V2, with the latter including the former as a subset. ChartQA is a recent open-domain chart Q&A dataset, evaluated on two subsets: augmented and human, where the augmented set is machine-generated and more extractive, while the human set is human-written and requires more complex reasoning.

**Chart-to-Text Task.** For the Chart-to-Text task, we used the C2T [48] dataset, which contains two subsets: Pew and Statista. The Pew subset consists of chart images from Pew Research Center with automatically extracted summaries, while the Statista subset consists of chart images from Statista with human-written summaries. This dataset provides a diverse set of chart styles and textual summaries, enabling the development of effective chart-to-text models.

**Evaluation Metrics.** We adopted task-specific metrics in our study. The Chart-to-Table task employed the metrics used in ChartOCR [42] to ensure fairness. For ChartQA, we used standard accuracy with a relaxed correctness criterion [44, 45] that permits a maximum of 1% numerical error. Additionally, BLEU4 was used to evaluate the Chart-to-Text task.

### 4.2. Training Details for Each Task

In this section, we provide an overview of the training details for our method on different tasks. In training, we first independently train the chart component detection mod-
Table 2. Performance comparison of different methods on the ChartQA task. The table shows the results of our proposed models, TaPas (Ours) and T5 (Ours), on four datasets: FQA, DVQA, PlotQA, and ChartQA. Our T5-based model achieved state-of-the-art performance on all datasets, outperforming previous methods by a large margin. The relaxed correctness criterion permits a maximum of 5% numerical error. TF and TN stand for Test-Familiar and Test-Novel, respectively, for the DVQA dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>FQA</th>
<th>DVQA</th>
<th>PQA</th>
<th>ChartQA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Val1</td>
<td>val2</td>
<td>Test1</td>
<td>Test2</td>
</tr>
<tr>
<td>IMG+QUES [26]</td>
<td>59.41</td>
<td>57.14</td>
<td>-</td>
<td>56.04</td>
</tr>
<tr>
<td>PReFIL [27]</td>
<td>94.84</td>
<td>93.26</td>
<td>94.88</td>
<td>93.16</td>
</tr>
<tr>
<td>CRCT [36]</td>
<td>94.61</td>
<td>85.04</td>
<td>94.23</td>
<td>84.77</td>
</tr>
<tr>
<td>PlotQA [45]</td>
<td>-</td>
<td>-</td>
<td>57.99</td>
<td>59.54</td>
</tr>
<tr>
<td>TaPas [44]</td>
<td>90.32</td>
<td>90.43</td>
<td>89.52</td>
<td>89.57</td>
</tr>
<tr>
<td>V-TaPas [44]</td>
<td>91.46</td>
<td>91.45</td>
<td>90.68</td>
<td>90.64</td>
</tr>
<tr>
<td>T5 [44]</td>
<td>87.97</td>
<td>87.83</td>
<td>87.56</td>
<td>87.57</td>
</tr>
<tr>
<td>VL-T5 [44]</td>
<td>88.60</td>
<td>88.49</td>
<td>88.20</td>
<td>88.18</td>
</tr>
<tr>
<td>TaPas (Ours)</td>
<td>91.12</td>
<td>91.40</td>
<td>91.15</td>
<td>91.25</td>
</tr>
<tr>
<td>T5 (Ours)</td>
<td>95.50</td>
<td>95.80</td>
<td>94.40</td>
<td>93.40</td>
</tr>
</tbody>
</table>

Table 3. Comparison of ChartReader with previous state-of-the-art methods on EC400K for Chart-to-Table task. The reported GPU hrs refer to the total amount of GPU processing time used for evaluating on 8 × A6000 GPUs, which is roughly equivalent to 8 times the running time, including time for IO and data preprocessing. Bold denotes the best performance.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Bar</th>
<th>Pie</th>
<th>Line</th>
<th>GPU hrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision [52]</td>
<td>0.58</td>
<td>0.84</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Faster-RCNN [41, 11]</td>
<td>0.80</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Rotation-RNN [41]</td>
<td>-</td>
<td>0.80</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ChartOCR [42]</td>
<td>0.92</td>
<td>0.92</td>
<td>0.96</td>
<td>57h</td>
</tr>
<tr>
<td>Ours</td>
<td>0.95</td>
<td>0.95</td>
<td>0.97</td>
<td>22h</td>
</tr>
</tbody>
</table>

4.3. State-of-the-Art Comparisons

Results of Chart-to-Table Task. In Table 3, we report the results of our proposed method on the EC400K dataset [42] for the Chart-to-Table task. Our approach achieved state-of-the-art performance, surpassing previous methods such as RotationRNN and Faster-RCNN, which rely solely on image recognition and object detection. The comparison highlights the superiority of our key point detection approach over bounding box detection. Our approach also outperforms earlier attempts such as Revision and ChartOCR, which heavily rely on image recognition overlay. ChartOCR represents the current state-of-the-art in the Chart-to-Table task. Our superior performance can be attributed to two factors. Firstly, we eliminated the need for heuristic rules and learned to recognize chart components by grouping center points and key points. Secondly, the recognized chart components are further utilized in subsequent chart-to-table and chartQA tasks, allowing our model...
to better understand the structure and semantic information of charts, leading to improved numerical evaluation performance. Moreover, our method achieved this superior performance with significantly less GPU hours compared to ChartOCR, as shown in Table 3. Specifically, our method only required 22 GPU hours, while ChartOCR used 57 GPU hours. This indicates that our method not only outperforms previous approaches but also does so with more efficient use of computing resources.

Results of ChartQA Task. Table 2 shows the results of our proposed models, TaPas (Ours) and T5 (Ours), on all datasets in the ChartQA task. Our approach utilizes the structural and semantic information of charts to answer questions, making it highly suitable for complex chart understanding tasks such as ChartQA. Specifically, our T5-based model outperformed the previous state-of-the-art method, PReFIL, on the FQA dataset by a small margin, and achieved state-of-the-art performance on the other three datasets. Our model benefits from joint training on chart comprehension and extraction tasks, enabling it to better understand the semantic and structural information of charts. Our T5-based model achieved state-of-the-art performance on the DVQA and PlotQA datasets, outperforming previous methods, including the current state-of-the-art method, V-TaPas, on both test sets. Additionally, our T5-based model achieved state-of-the-art performance on the ChartQA dataset, outperforming previous methods by a large margin, indicating good generalization to unseen data.

Table 4. Results of Chart-to-Text task on the Pew and Statista datasets. Our T5-based model achieved state-of-the-art performance on both datasets, outperforming previous state-of-the-art methods.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Pew</th>
<th>Statista</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5 [40]</td>
<td>10.5</td>
<td>35.3</td>
</tr>
<tr>
<td>PaLI-17B (res. 224) [40]</td>
<td>10.0</td>
<td>40.2</td>
</tr>
<tr>
<td>PaLI-17B (res. 588) [40]</td>
<td>11.2</td>
<td>41.4</td>
</tr>
<tr>
<td>Pix2Struct [35]</td>
<td>10.3</td>
<td>38.0</td>
</tr>
<tr>
<td>MATCHA [40]</td>
<td>12.2</td>
<td>39.4</td>
</tr>
<tr>
<td>T5 (Ours)</td>
<td>14.2</td>
<td>44.2</td>
</tr>
</tbody>
</table>

5. Ablation Study

Chart Component Detection. We compare our full model with two variants that lack key point detection or group detection. The full model outperforms the ablated models on all three chart types, and key point detection improves performance on line charts, while the group module is more effective for bar charts. These results highlight the importance of both components for accurate chart component detection.

Table 5. Ablation study on chart component detection, comparing the performance of the full model with two variants that do not have either key point detection or group detection. The full model outperforms the ablated models on all three chart types, highlighting the importance of both components for accurate chart component detection.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Bar</th>
<th>Pie</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o Key Point</td>
<td>0.83</td>
<td>0.73</td>
<td>0.63</td>
</tr>
<tr>
<td>w/o Group</td>
<td>0.77</td>
<td>0.81</td>
<td>0.52</td>
</tr>
<tr>
<td>Ours</td>
<td>0.95</td>
<td>0.95</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Input Encoding. We conducted an ablation study on input encoding methods, comparing full model performance with different ablated versions on PlotQA-V1 and PlotQA-V2 datasets. Table 6 shows that the full model outperforms all ablated versions, with location and appearance embeddings contributing the most to performance. This indicates that spatial and visual information is crucial for chart comprehension tasks.

Table 6. Ablation study on input encoding for the PlotQA-V1 and PlotQA-V2 datasets. The table shows that the full model outperforms all ablated versions, and the location and appearance embeddings contribute the most to the overall performance.

<table>
<thead>
<tr>
<th>Ablation Token</th>
<th>PlotQA-V1</th>
<th>PlotQA-V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o type</td>
<td>60.20</td>
<td>48.20</td>
</tr>
<tr>
<td>w/o location</td>
<td>64.20</td>
<td>51.20</td>
</tr>
<tr>
<td>w/o appearance</td>
<td>59.10</td>
<td>45.20</td>
</tr>
<tr>
<td>w/o CCD</td>
<td>52.30</td>
<td>40.20</td>
</tr>
<tr>
<td>Ours (TaPas)</td>
<td>74.20</td>
<td>56.20</td>
</tr>
</tbody>
</table>

Impact of Hyperparameter $\alpha$ on Data Variable Replacement. We conducted an ablation study to determine the optimal value of $\alpha$ in the Data Variable Replacement Technique. As shown in Table 5, the optimal $\alpha$ varied from 0.2 to 0.7 depending on the dataset complexity. Table 7 compares numerical value accuracy with and without data variable replacement ($\alpha$ set to 0 vs. optimal $\alpha$). Figure 5 shows the performance variation with $\alpha$ for the T5 model.

Multi-Task Training. We conducted ablation experiments to evaluate the effectiveness of a multi-task training approach on the ChartQA task. Our model, trained on the
The impact of the hyperparameter $\alpha$ on model performance in chart understanding. The experimental results indicate that the weight of the $\alpha$ parameter increases as the number of open-ended questions grows, as the model needs to use variable replacement techniques more frequently to avoid errors caused by random guessing of unknown markers. Appropriate selection of the $\alpha$ parameter is crucial for achieving high performance and training efficiency of pretrained models.

### Table 7. Impact of Hyperparameter $\alpha$ in Data Variable Replacement Technique

<table>
<thead>
<tr>
<th></th>
<th>FQA</th>
<th>DVQA</th>
<th>PlotQA</th>
<th>CharQA</th>
<th>C2T</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>0.3</td>
<td>0.2</td>
<td>0.6</td>
<td>0.7</td>
<td>0.5</td>
</tr>
<tr>
<td>Rate</td>
<td>5%</td>
<td>5%</td>
<td>20%</td>
<td>40%</td>
<td>30%</td>
</tr>
</tbody>
</table>

C4+ pre-training dataset with all three chart-related tasks, achieved the highest performance on both validation and test sets, as demonstrated in Table 9. These results highlight the significance of incorporating multi-task training.

### Table 9. Results of Ablation Study on the Effect of Pre-Training Dataset on the ChartQA Task

<table>
<thead>
<tr>
<th>Model</th>
<th>PT</th>
<th>EF</th>
<th>MT</th>
<th>Val</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5</td>
<td>C4+(PQA)</td>
<td>CQA</td>
<td>✓</td>
<td>38.4</td>
<td>39.2</td>
</tr>
<tr>
<td>T5</td>
<td>C4+(PQA)</td>
<td>CQA</td>
<td>✓</td>
<td>42.3</td>
<td>41.5</td>
</tr>
<tr>
<td>T5</td>
<td>C4+(PQA+CQA)</td>
<td>CQA</td>
<td>✓</td>
<td>39.2</td>
<td>39.5</td>
</tr>
<tr>
<td>T5</td>
<td>C4+(PQA+CQA)</td>
<td>CQA</td>
<td>✓</td>
<td>45.5</td>
<td>43.2</td>
</tr>
<tr>
<td>T5</td>
<td>C4+(PQA+CQA+C2T)</td>
<td>CQA</td>
<td>✓</td>
<td>41.2</td>
<td>42.2</td>
</tr>
<tr>
<td>T5</td>
<td>C4+(PQA+CQA+C2T)</td>
<td>CQA</td>
<td>✓</td>
<td>49.5</td>
<td>52.6</td>
</tr>
</tbody>
</table>

### Table 8. Ablation for Data Variable Replacement on FQA Val1, DVQA TF, PQA TestV1, ChartQA Val for ChartQA task, and Pew metric for C2T on Chart-to-Text

<table>
<thead>
<tr>
<th></th>
<th>FQA</th>
<th>DVQA</th>
<th>PQA</th>
<th>ChartQA</th>
<th>C2T</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaPas ($\alpha = 0$)</td>
<td>90.5</td>
<td>86.6</td>
<td>58.6</td>
<td>41.2</td>
<td>10.2</td>
</tr>
<tr>
<td>TaPas (optimal $\alpha$)</td>
<td>91.1</td>
<td>92.2</td>
<td>74.2</td>
<td>48.3</td>
<td>12.8</td>
</tr>
<tr>
<td>T5 ($\alpha = 0$)</td>
<td>92.8</td>
<td>91.3</td>
<td>74.5</td>
<td>43.2</td>
<td>11.6</td>
</tr>
<tr>
<td>T5 (optimal $\alpha$)</td>
<td>95.5</td>
<td>95.4</td>
<td>78.1</td>
<td>49.5</td>
<td>14.2</td>
</tr>
</tbody>
</table>

6. Conclusion

In this paper, we propose ChartReader, a framework that integrates chart derendering and comprehension tasks using a transformer-based chart component detection module and a pre-trained vision-language model. Our approach enhances accuracy and eliminates manual rule-making. Through experiments, we outperform existing methods in Chart-to-Table, ChartQA, and Chart-to-Text tasks. Our framework reduces manual effort in chart analysis and facilitates a universal chart comprehension model.

## Acknowledgement

We extend our sincere gratitude to Siyao Li, Jingdong Sun, and Teruko Mitamura for their invaluable contributions to this paper. This work received support from the Air Force Research Laboratory under agreement number FA8750-19-2-0200, the Defense Advanced Research Projects Agency (DARPA) grants funded through the GAILA program (award HR00111990063), and the AIDA program (FA8750-18-20018).

The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.
References


