Advancing Chart Question Answering with Robust Chart Component Recognition

Hanwen Zheng
Virginia Tech
zoez@vt.edu

Sijia Wang
Virginia Tech
sijiawang@vt.edu

Chris Thomas
Virginia Tech
christhomas@vt.edu

Lifu Huang
University of California, Davis
lfuhuang@ucdavis.edu

Abstract

Chart comprehension presents significant challenges for machine learning models due to the diverse and intricate shapes of charts. Existing multimodal methods often overlook these visual features or fail to integrate them effectively for Chart Question Answering. To address this, we introduce CHARTFORMER*, a unified framework that enhances chart component recognition by accurately identifying and classifying components such as bars, lines, pies, titles, legends, and axes. Additionally, we propose a novel Question-guided Deformable Co-Attention (QDCAt) mechanism, which fuses chart features encoded by* CHARTFORMER *with the given question, leveraging the question's guidance to ground the correct answer. Extensive experiments demonstrate a 3.2% improvement in mAP over the baselines for chart component recognition. For ChartQA and OpenCQA tasks, our approach achieves improvements of 15.4% in accuracy and 0.8 in BLEU score, respectively, underscoring the robustness of our solution for detailed visual data interpretation across various applications.*[1]

1. Introduction

Comprehending charts and correctly answering chart-related questions [4, 18, 28] is essential in today's data-driven world. Charts are powerful tools to distill complex data into visual formats, making it easier to identify trends, patterns, and insights at a glance. Despite the significant research progress on Visual Question Answering [1, 9, 33, 35, 38, 41, 42, 44], Chart Question Answering is particularly challenging as it requires seamless and fine-grained interpretation and analysis of both textual and visual elements in the charts to answer natural language questions [4, 16, 18, 21, 28–30, 49]. Consider the example shown in Figure 1. To correctly answer the question based on the given chart, models need to accurately locate textual elements such as "*White*" in the category axis and "*Female presidents*" in the legend. In addition, they also need to identify the visual elements such as the blue bar for "*White*" category and correctly link it to the corresponding text label, i.e., "*33%*".

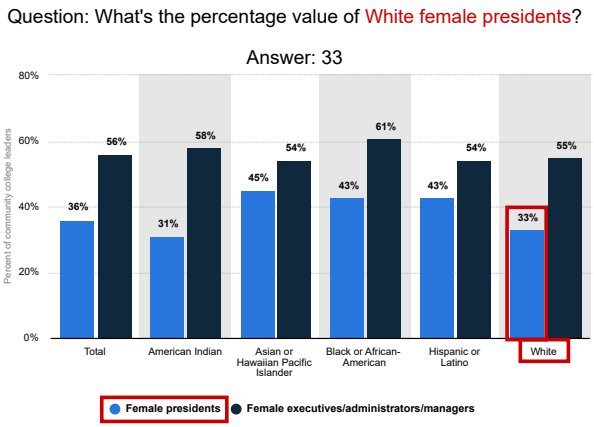

Figure 1. An example of Chart Question Answering.

Though many efforts have been made on chart component recognition to correctly identify key components such as chart types (e.g., *pie*, *bar*, and *line* charts), visual elements in each plot (e.g., *connected lines*, *colors*), and textual elements (e.g., *legends* and *value axes*), they usually require a pipeline approach that first detects key points of lines or boxes and then classifies and groups the detected regions. Such methods struggle to comprehend complex graphics, such as stacked boxes, overlapped labels, and crossed lines. For chart question answering, it is essential to incorporate rich and accurate chart information and ensure proper attention is given to the relevant chart regions based on the specific questions. Existing approaches that leverage OCR tools or extracted tables often introduce noise and confusion due to inaccuracies. Additionally, existing models typically fuse the visual information with the question embedding at a later stage, relinquishing the ability to guide attention to the relevant parts of the graphics via the question.

To tackle these challenges, we propose a novel framework named QDCHART, which integrates an innovative

[1]The source code and dataset are publicly available at https://github.com/VT-NLP/chartQA

unified solution, CHARTFORMER, to recognize all the various chart components from diverse types of charts. CHARTFORMER leverages deformable attention [40] to effectively capture the visuals of chart graphics, and incorporates a mask attention mechanism [3] with learnable query features to cover diverse chart components. To address chart question answering tasks, we propose a novel Question-guided Deformable Co-Attention (QDCAt) mechanism that leverages the rich chart features encoded by CHARTFORMER. This mechanism fuses question information with chart features through a Question-guided Offset Network and integrates visual and chart-related features using a deformable co-attention module. The resulting question-guided features are then passed to a text decoder to generate the answer to the given question.

We first evaluate the effectiveness of CHARTFORMER on chart component recognition, on the public benchmark dataset ExcelChart400K [26]. CHARTFORMER significantly outperforms existing strong baselines, such as DAT [40] Mask2former [3], by 11.4% and 3.2% in mAP, respectively. It especially shows superior performance on stacked bars, overlapped or fluctuating lines, and narrow pie slices. We further evaluate QDCHART on ChartQA [28] and OpenCQA [15]. QDCHART outperforms the previous strong baseline UniChart [29] by 0.5% in accuracy and surpasses Donut [16] by 0.8 in BLEU score. Our model excels in handling visually related questions, particularly those involving color disambiguation or size comparison of chart components. Additionally, it demonstrates superior ability in effectively grounding key chart information.

We summarize the contribution of this work as follows:

- CHARTFORMER, a state-of-the-art end-to-end model for unified chart component recognition, provides accurate identification for diverse class objects with distinct visual structures and rich chart semantics.
- QDCHART, a multimodal application leveraging CHARTFORMER's rich chart semantics with a novel Question-guided Deformable Co-Attention (QDCAt) fusion layer for ChartQA, enables the model to focus on components related to the question.
- CHARTFORMER exceeds the baseline model by 3.2% in mAP on chart component recognition, while QDCHART surpasses the baseline models by 15.4% in accuracy on ChartQA and 0.8 in BLEU score on OpenCQA.
- An automatically annotated unified chart component recognition instance segmentation dataset featuring several prominent chart types.

2. Related Work

Chart Component Recognition Existing approaches for chart element recognition can be divided into three broad categories: detection via bounding boxes [6, 23, 27], key point detection and grouping [4, 26, 27, 43], and line graph detection [17, 31]. Among these, ChartOCR [26] uses an hourglass network to identify key points and a rule-based approach to associate them with chart elements. Lenovo [27] trains separate detectors for points and bars, followed by a deep neural network to measure feature similarities for data conversion. ChartDETR [43] employs a transformer-encoder-decoder model to detect and categorize key point groups within a unified framework. Lineformer [17] focuses on line charts, treating line detection as instance segmentation using the Mask2former model [3]. Compared to previous studies, our proposed CHARTFORMER instead employs a comprehensive end-to-end instance segmentation framework to enhance chart component recognition.

Object Detection and Instance Segmentation Object detection and instance segmentation [8,45,48] are core tasks in Computer Vision and have been widely explored by using Convolutional Neural Networks [12, 13, 36, 37, 39]. Recently, many new approaches have been developed based on Transformer [46, 47], inspired by its success in the field of Natural Language Processing. Among them, ViT [10] processes images as non-overlapping patch sequences, using global attention to capture long-range dependencies. Swin Transformer [25], on the other hand, employs partitioned window attention to focus on specific regions within images. Enhanced attention mechanisms like the Deformable Convolutional Network (DCN) [7] and Masked attention Mask Transformer (Mask2former) [3] are further introduced to enhance these approaches. DAT [40] improves DCN's capabilities by learning deformed key points through feature sampling and updating positional embeddings with relative position bias. Our work leverages the strengths of instance segmentation frameworks on small and varied objects to improve chart understanding.

Chart Question Answering Recent advancements in language and multimodal models have significantly enhanced their ability to tackle the complex reasoning required for ChartQA tasks. Donut [16] is a vision-encoder-text-decoder model that leverages Swin Transformer [25] and MBart [24] to answer questions with a visual context. VL-T5 [5] extends [34] by incorporating visual features from chart images, while VisionTaPas [28] extends TaPas [14] by integrating a vision transformer encoder to process chart images. ChartT5 [49] improves chart understanding by leveraging a visual and language pre-training framework on chart images and predicted table pairs. ChartReader [4] leverages a transformer-based model to detect chart components and Pix2Struct [18] adopts an image-encoder-text-decoder architecture and introduces a screenshot parsing pre-training objective based on the HTML source of web pages, aiming to enhance the model's layout and language understanding

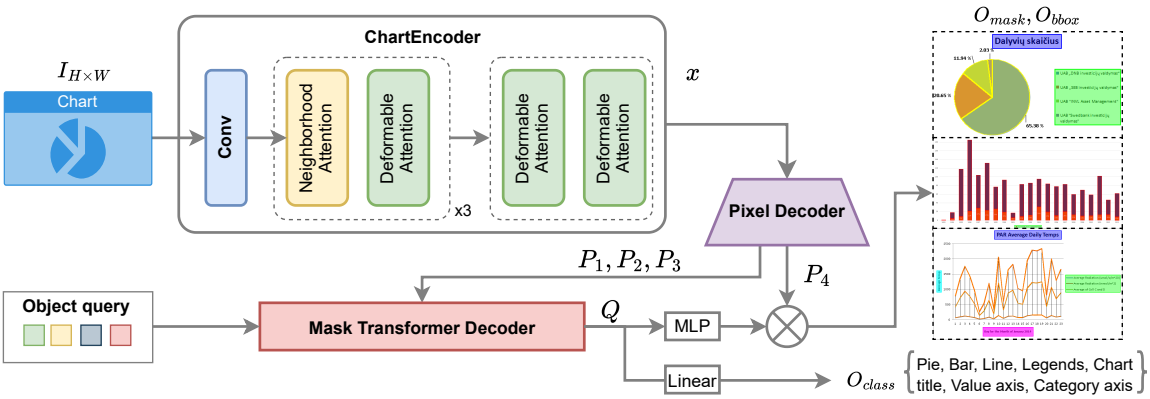

Figure 2. Model architecture of CHARTFORMER for chart component recognition

capabilities. MatCha [21] utilizes two pre-training tasks, chart derendering, and math reasoning, to enhance visual language understanding. Similarly, UniChart [29] enhances Donut's capabilities by pre-training on four chart-related tasks: data table generation, numerical & visual reasoning, open-ended question answering, and chart summarization. DEPLOT [20] relies on plot-to-text translation, followed by reasoning over the translated text. Compared to these existing efforts, QDCHART employs a novel unified solution for detecting all the chart components and further enhances the visual reasoning capability by attending to the components that are particularly related to questions.

3. Method

In this section, we first introduce CHARTFORMER, the first unified and end-to-end solution for recognizing components from diverse types of charts with distinct visual structures (Section 3.1), and then illustrate QDCHART which integrates the chart components identified by CHARTFORMER and selectively incorporates them to answer the target question (Section 3.2).

3.1. CHARTFORMER

As shown in Figure 2, given an input chart image $\boldsymbol{I}$ with dimensions $H \times W$, **chart component recognition** aims to identify and classify various components within the chart, including both visual elements such as *lines*, *bars*, and *pie slices*, and textual elements such as *value axes*, *category axes*, *legends*, and *chart titles*. Let C be the set of all chart component types, e.g., $C = \{\text{"Bar"}, \text{"Line"}, \text{"Pie"}, \ldots\}$. Let M_c represent the set of instance segmentation regions corresponding to a particular chart component type $c \in C$, and $M = \{M_c\}_{c \in C}$ be the set of all annotations for the image $\boldsymbol{I}$.

CHARTFORMER is designed to learn to accurately identify and classify each chart component in chart images, and consists of three main modules: a vision encoder, a pixel decoder, and a mask transformer decoder. Formally, the input chart image $\boldsymbol{I} \in \mathbb{R}^{H \times W}$ first undergoes processing by a CNN layer to generate an initial feature map $\boldsymbol{E} \in \mathbb{R}^{K \times H/4 \times W/4}$ with a channel size $K = 64$ in our experimental setting. This feature map $\boldsymbol{E}$ is fed into the **vision encoder** for further processing and feature learning. The vision encoder features multiple blocks of neighborhood attention [11] and deformable attention [40]. Neighborhood attention expands each pixel's attention span to its nearest neighborhood

$$\boldsymbol{\chi} = \text{NeigAttention}(\boldsymbol{E}) \,. \tag{1}$$

Chart images have distinct borders and consecutive visual elements like lines and bars, thus applying deformable attention becomes intuitive to emphasize precise focus on the visual connections. In deformable attention, uniformly spaced reference points p are offset by Δp, obtained via an offset network θ_{offset} applied to the query vectors $\boldsymbol{q}$. Features are then computed using bilinear sampling ϕ at the deformed points

$$\Delta p = \theta_{\text{offset}}(\boldsymbol{q}), \quad \tilde{\boldsymbol{\chi}} = \phi(\boldsymbol{\chi}; p + \Delta p) \,. \tag{2}$$

Then the deformation attention is computed as

$$\boldsymbol{q} = \boldsymbol{\chi} W_q \,, \quad \boldsymbol{k} = \tilde{\boldsymbol{\chi}} W_k \,, \quad \boldsymbol{v} = \tilde{\boldsymbol{\chi}} W_v \,, \tag{3}$$

$$\text{DefoAttention}(\boldsymbol{\chi}) = \text{softmax}\left(\boldsymbol{q}\boldsymbol{k}^\top / \sqrt{d}\right) \boldsymbol{v} \,, \tag{4}$$

where key $\boldsymbol{k}$ and value $\boldsymbol{v}$ vectors are projected from sampled features $\tilde{\boldsymbol{\chi}}$, and d is the dimension of the attention head.

These specialized attention mechanisms allow CHARTFORMER to effectively capture local and global contextual information, thereby enhancing the ability to extract meaningful features from chart images. Together, the CNN layer and the vision encoder extract features $\boldsymbol{x} \in \mathbb{R}^{8K \times H/32 \times W/32}$ from the input image $\boldsymbol{I}$

$$\boldsymbol{x} = \mathcal{E}^{\theta}_{\text{ChartEncoder}}(\boldsymbol{I}) \,. \tag{5}$$

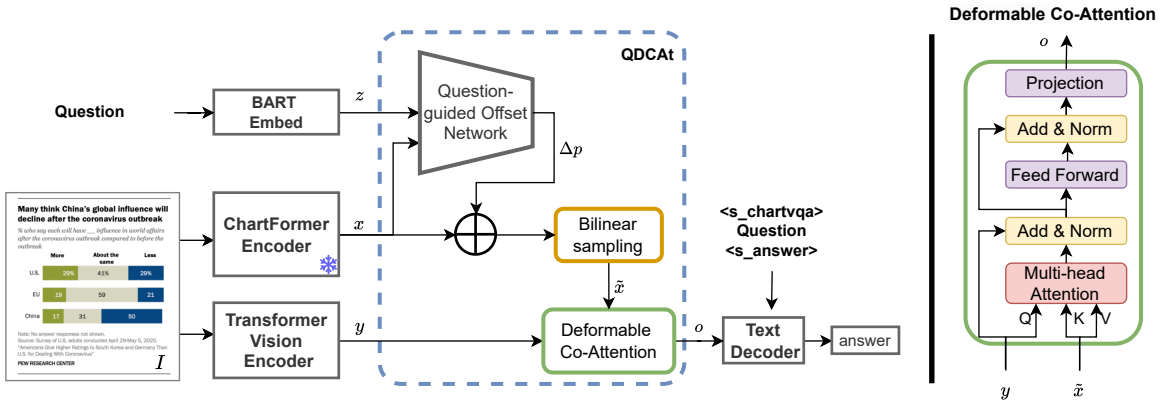

Figure 3. The QDCHART model structure

The **pixel decoder** then upsamples the extracted features $\boldsymbol{x}$ and outputs a 2D per-pixel embedding $\boldsymbol{P} \in \mathbb{R}^{C_E \times HW}$, where $C_E = 256$ is the number of channels

$$\boldsymbol{P} = \boldsymbol{P}_4 \,, \quad \boldsymbol{P}_i = \mathcal{D}^i_{\text{pixel}}(\boldsymbol{P}_{i-1}) \,, \quad \boldsymbol{P}_0 = \boldsymbol{x} \,. \tag{6}$$

Subsequently, a **mask transformer decoder** combines object queries $\boldsymbol{u} \in \mathbb{R}^{C_Q \times N}$ and pixel decoder features to compute embeddings $\boldsymbol{Q} \in \mathbb{R}^{C_Q \times N}$, where $C_Q = 256$ denotes the number of channels and N denotes the number of object queries

$$\boldsymbol{Q} = \mathcal{D}_{\text{mask}}(\boldsymbol{u}; \boldsymbol{P}_{1:3}) \,. \tag{7}$$

The embeddings $\boldsymbol{Q}$ are passed through dense layers to predict object classes $\boldsymbol{O}^i_{\text{class}} \in \{\Delta^L\}_{i=1}^N$ and binary per-pixel mask predictions $\boldsymbol{O}_{\text{mask}} \in \{0,1\}^{N \times HW}$

$$\boldsymbol{O}^i_{\text{class}} = \text{softmax}(\boldsymbol{Q}_i^\top W) \,, \tag{8}$$

$$\boldsymbol{O}_{\text{mask}} = s\Big(\sigma\big(\boldsymbol{M}^\top \boldsymbol{P}\big) - t\Big) \,, \tag{9}$$

where σ is the element-wise sigmoid function, $t \in [0,1]$ is a threshold parameter, s is the step function, and $\boldsymbol{M} = \text{MLP}(\boldsymbol{Q})$. The bounding box $\boldsymbol{O}_{bbox}$ can be directly obtained by drawing the smallest box that encloses the segmentation mask. Following [3], we use a combination of focal loss and dice loss for $\boldsymbol{O}_{\text{mask}}$, and classification loss for $\boldsymbol{O}_{\text{class}}$.

3.2. QDCHART

Given an input image $\boldsymbol{I}$ and a natural language question Q, we further design QDCHART to provide an answer A by leveraging the chart components detected by CHARTFORMER. As shown in Figure 3, QDCHART consists of four main modules: a chart encoder, a vision encoder, a Question-guided Deformable Co-Attention (QDCAt) fusion block that fuses the output of two encoders, and a text decoder.

Chart Encoder We include the vision encoder of the CHARTFORMER model ($\mathcal{E}^\theta_{\text{ChartEncoder}}$) as a primary image encoder, utilized to capture explicit chart segment information. During the fine-tuning stage on the ChartQA dataset, we freeze the weights θ of the CHARTFORMER model to reduce training costs.

Vision Encoder The Vision Encoder module follows the previous work [16] and utilizes a Swin Transformer [25] architecture to provide complementary visual information, denoted as $\mathcal{E}^\theta_{\text{SWIN}}(\boldsymbol{I})$. The parameters are initialized based on the pre-trained model [16]. We pass the image through the Swin encoder to obtain the feature $\boldsymbol{y}$

$$\boldsymbol{y} = \mathcal{E}^\theta_{\text{SWIN}}(\boldsymbol{I}) \,, \tag{10}$$

where θ are trainable parameters of the Vision Encoder.

Question-guided Deformable Co-Attention To fuse image features from complimentary encoders, and to incorporate information guided by the question, we propose Question-guided Deformable Co-Attention (**QDCAt**), which consists of a question-guided offset network and a deformable co-attention block.

To capture complex spatial relationships and patterns in images and associate them with VQA questions, a Question-guided Offset Network (**QON**) is proposed. The proposed QON θ^z_{offset} is distinct in being conditioned on the question embedding provided in the ChartQA task, as shown in Figure 3. We obtain the token embeddings of the question as provided by the last hidden layer of MBart [24] and denote the output as $\boldsymbol{z}$. The input tensor $\boldsymbol{x}$ is passed through a projection layer[2] parametrized by the weights W_a to obtain $\boldsymbol{a} = \boldsymbol{x}W_a$. We take the dot product of $\boldsymbol{z}$ and $\boldsymbol{a}$ to get $\boldsymbol{z}\boldsymbol{a}^\top$, which we pass into a $k \times k$ convolution layer Conv, normalization Norm, GELU activation, and a projec-

[2] "Projection layer" refers to a convolution layer with a 1×1 kernel.

tion layer parametrized by the weights $W_{\Delta p^z}$ to obtain

$$\begin{aligned}\Delta p^z &= \theta^z_{\text{offset}}(\boldsymbol{x}) \\ &= \text{GELU}(\text{Norm}(\text{Conv}(\boldsymbol{z}(\boldsymbol{x}W_a)^\top)))W_{\Delta p^z}.\end{aligned} \tag{11}$$

The idea behind giving the offset network the combined input of $\boldsymbol{z}$ and $\boldsymbol{a}$ is to make the sampled points align with image locations with semantic significance for the question of the particular data sample, as opposed to locations with overall significance. This allows further features learned by the model to represent visual queues with specific relevance to the given question. A visual representation of the question-guided offset network is provided in Figure 4 as well as Figure 9 in Supp. D.

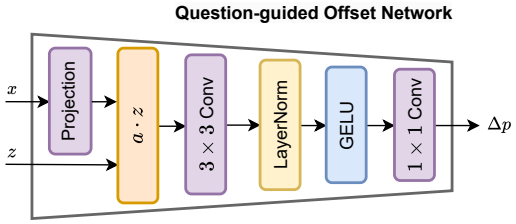

Figure 4. Question-guided offset network

The question-guided offsets Δp^z are then used to offset per-pixel coordinates p, positioning the points used for bilinear sampling

$$\tilde{\boldsymbol{x}} = \phi(\boldsymbol{x}; p + \Delta p^z)\,. \tag{12}$$

Next, we introduce the **deformable co-attention block**. This novel attention block integrates and enhances the advantages of co-attention and deformable attention, where co-attention combines features extracted from multiple modalities, and deformable attention in formula (4) emphasizes attention computed at semantically relevant features. The proposed *deformable co-attention* extends this approach by combining features $\tilde{\boldsymbol{x}}$ and $\boldsymbol{y}$.

$$\boldsymbol{q} = \boldsymbol{y}W_q\,, \quad \boldsymbol{k} = \tilde{\boldsymbol{x}}W_k\,, \quad \boldsymbol{v} = \tilde{\boldsymbol{x}}W_v\,, \tag{13}$$

$$\text{DCAttention}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = \text{softmax}\left(\boldsymbol{q}\boldsymbol{k}^\top/\sqrt{d}\right)\boldsymbol{v}\,. \tag{14}$$

The output of $\text{DCAttention}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ is passed through residual connections, normalization layers, and a feed-forward network. The Add and LayerNorm operations perform element-wise addition of features and normalization. Their first occurrence surrounds deformable co-attention as follows:

$$\boldsymbol{b} = \text{LayerNorm}(\boldsymbol{y} + \text{DCAttention}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}))\,. \tag{15}$$

The feed-forward network consists of a 3×3 convolution layer followed by ReLU activation. The second occurrence of Add and LayerNorm surrounds the feed-forward network

$$\boldsymbol{c} = \text{LayerNorm}(\boldsymbol{b} + \text{ReLU}(\text{CNN}(\boldsymbol{b})))\,. \tag{16}$$

The output $\boldsymbol{c}$ is lastly projected via W_o to obtain $\boldsymbol{o} = \boldsymbol{c}W_o$, which is the output of the deformable co-attention block as the final image feature.

Text Decoder Following the approach of Donut [16], we also employ the MBart decoder [24] for answer output generation. Task-specific prompts, for instance, `<chartvqa>`, `<s_answer>` are provided to the decoder as special tokens, and the decoder generates the output by predicting the next token based on the prompted context. The text decoder, denoted as $\mathcal{D}^\theta_{\text{text}}$, takes the processed feature o by QDCAt and the task-specific prompt p to generate answers A

$$A = \mathcal{D}^\theta_{\text{text}}(p; \boldsymbol{o})\,. \tag{17}$$

4. Experimental Setup

We first evaluate CHARTFORMER on the chart component recognition task and then further assess the effectiveness of QDCHART on chart question answering tasks.

4.1. Chart Component Recognition

Baselines We employ four advanced models as strong baselines for chart component recognition and compare them with CHARTFORMER: **Mask R-CNN**, which [12] extends Faster R-CNN by adding a branch for predicting segmentation masks; **SOLOv2** [39] utilized a specialized segmentation branch with decoupled head for better mask feature learning; **Mask2former** [3]incorporates masked attention mechanisms for unified segmentation tasks; **DAT** [40], characterized by its integration of deformable attention mechanisms and global image features, facilitates the comprehensive analysis of multiple chart components.

Dataset Previous chart component recognition datasets are limited in their coverage of chart element types, making them inadequate for a comprehensive evaluation and only offering restricted chart understanding analysis for subsequent ChartQA tasks. Thus we leverage the ExcelChart400K dataset to automatically create annotations for 7 categories, including bar, line, pie, legend, value axis title, chart title, and category axis title[3]. As the annotations in the ExcelChart400K dataset are formatted for keypoint detection, we convert them to an instance segmentation format. Each object and its category remains the same, but we now create segmentation masks from the given key points. We store segmentation masks in the format of *polygons*, meaning that for each chart component, we create a collection $\mathcal{P} = [x_1, y_1, \ldots, x_n, y_n]$ defining the adjacent vertices (x_i, y_i) of the associated polygon. The procedure for converting these annotations to an instance segmentation format is outlined in Supp. A. Details about the category distribution of the annotations can be found in Table 1.

[3]Category axis title refers to axis title for categorical values.

Split	Train		Validation		Test	
Bar	1,126,919	71.28%	46,115	77.69%	49,409	78.92%
Line	92,373	5.84%	2,466	4.15%	2,467	3.94%
Pie	141,449	8.95%	3,955	6.66%	3,775	6.03%
Legend	75,534	4.78%	2,503	4.22%	2,564	4.10%
ValueAxisTitle	47,704	3.02%	1,446	2.44%	1,417	2.26%
ChartTitle	69,289	4.38%	2,127	3.58%	2,200	3.51%
CategoryAxisTitle	27,740	1.75%	745	1.26%	773	1.23%
Total	1,581,008	100.00 %	59,357	100.00 %	62,605	100.00 %

Table 1. Category distribution in the ExcelChart400K instance segmentation dataset.

Evaluation Metrics We evaluate the performance of chart component recognition with three metrics: (1) mAP (mean Average Precision) is a common metric for evaluating object detection and instance segmentation tasks. It calculates the average precision for each class across all recall levels and then averages these values to get a single score. (2) mAP50 specifically refers to the mean Average Precision calculated at an intersection over union (IoU) threshold of 0.5. (3) mAP75 is similar to mAP50 but calculated at an IoU threshold of 0.75, and thus a stricter metric requiring a higher amount of overlap.

4.2. Chart Question Answering

Baselines We benchmark our models against 13 established baselines to assess its performance: **T5** [34], a unified sequence-to-sequence Transformer model, well known for its state-of-the-art results in text-to-text tasks; **VL-T5** [5], which modifies T5 to address Vision Language tasks by generating text from multimodal inputs; **TaPas** [28] model and its extension, **VisionTapas** [28], recognized for their efficacy in table encoding, the latter has been adapted for chart-based question answering with an additional table input; in addition, we also compare our model with the pre-trained versions of **VisionTapas** and **VL-T5** on PlotQA [30] with chart VQA tasks; **ChartReader** [4], which is based on the T5 model and leverages extracted chart information as input; **Pix2Struct** [18], which facilitates pixel-to-text design for visually rich document understanding; **ChartT5** [49], which is pre-trained based on VL-T5 with a masked chart-table pairing objective using chart VQA data; two excessively pre-trained models with additional chart comprehension data: **Matcha** [21] and **UniChart** [29] and three large language models (LLM): **PaLI-17B** [2], **LLaVA1.5-13B** [22], and **DEPLOT** [20], a LLM for one-shot reasoning. **Donut** [16], an OCR-oriented vision-encoder-decoder model, is distinguished by its proficiency in vision-based question answering and summarization. These baselines were selected for their relevance and demonstrated effectiveness in their respective domains, providing a comprehensive reference for comparing our model's capabilities.

Dataset The ChartQA dataset [28] includes three prevalent chart types – a total of $18,337$ bar charts, $2,800$ line charts, and 808 pie charts. The dataset consists of two subsets: a human-written question set and a machine-generated question set. Details about the distribution of data in the ChartQA dataset can be found in Table 6 in Supp. D. OpenCQA [15] is a challenging chart question answering dataset featuring open-ended questions about a chart with explanatory answers. A case study can be found in Supp. D.

Evaluation Metrics To evaluate our approach, we follow previous works [26, 28] and utilize Relaxed Accuracy (RA) for ChartQA. Relaxed Accuracy allows a minor 5% inaccuracy of numeric answers to account for errors in the data extraction process. For non-numerical answers, Relaxed Accuracy requires an exact match (ignoring case) for the answer to be counted as correct. For the OpenCQA dataset, we use BLEU score [19, 32] as the evaluation metric.

5. Results

5.1. Chart Component Recognition

Table 2 illustrates the performance of all the models employed in chart element detection and classification. Among them, CHARTFORMER exhibits the most impressive performance across all mAP evaluation metrics. Notably, the results attained by CHARTFORMER remain consistently superior across all categories, attaining either the top or second best mAP score, as shown in Table 3. Note that the line component emerges as the most challenging category due to its distinctiveness from other categories and its unique structural characteristics.

Model	mAP	mAP50	mAP75
Mask R-CNN [12]	48.5	62.9	54.8
SOLOv2 [39]	18.4	37.0	16.1
DAT [40]	53.5	64.3	59.0
Mask2former [3]	61.7	82.4	65.5
CHARTFORMER	**64.9**	**85.0**	**69.1**

Table 2. Experimental results on chart component recognition.

Compared to DAT, CHARTFORMER achieves better performance, particularly in categories that are challenging for DAT, such as lines. Experimental results indicate that the narrower the line width we annotate, the more difficult it is

Model	Bar	Line	Pie	Legend	ValueAT	ChartTitle	CategoryAT	Average
Mask R-CNN [12]	69.6	0.1	61.6	69.7	37.2	65.2	36.1	48.5
SOLOv2 [39]	3.1	0.0	33.9	29.1	10.9	42.7	9.3	18.4
DAT [40]	**75.0**	0.6	63.3	81.2	40.9	**71.8**	41.6	53.5
Mask2former [3]	63.1	28.8	79.8	80.3	52.8	69.6	**57.7**	61.7
CHARTFORMER	73.1	**36.3**	**84.0**	**81.9**	**53.1**	71.6	54.2	**64.9**

Table 3. Chart component recognition results (mAP) for all categories. AT denotes AxisTitle. (Best performances are highlighted in bold.)

for DAT to predict the line segments accurately. However, CHARTFORMER consistently predicts the line segments accurately, regardless of how narrow the line annotations are. Additionally, compared to Mask2former, CHARTFORMER increases the accuracy for bar segments by 10.0%. In complex scenarios such as stacked bar charts and overlapping line charts, CHARTFORMER more accurately detects the correct number of components. Detailed comparison examples are provided in Figures 12, 13, and 11 in Supp. D. These examples reveal the shortcomings of competing models, such as the inability to predict smooth pie edges, small bars, or steep lines. Though Mask2former performs relatively well, it occasionally misses stacked bar predictions and overcompensates on the width of the lines.

5.2. Chart Question Answering

The results of the chart visual question answering task are summarized in Table 4. UniChart is pre-trained on 13 million high-resolution chart question-answering data instances, while Donut has no chart-related pre-training. We integrate our QDCAT approach with both UniChart and Donut model weights, resulting in QDCHART-UniChart and QDCHART-Donut. The experiments show that QDCHART outperforms all non-pretrained and pre-trained baseline methods, as well as some LLM-based baselines. It is important to note that UniChart has been pre-trained extensively with ChartQA data but not with additional open-ended chart question-answering data, which accounts for its high performance on the ChartQA task and lower performance on the OpenCQA task. In contrast, QDCHART-Donut undergoes minimal pre-training with a pseudo-OCR task, which may not capture the same level of detail and complexity for the ChartQA task, leading to lower performance. The results demonstrate that our QDCAT approach improves performance on both tasks and suggest that visual reasoning may be more effective than pre-training on the OpenCQA task.

We further visualize deformed points sampled by the bilinear sampling step in QDCHART in Figure 5 to demonstrate the effectiveness of the proposed framework. We present three human-written ChartQA examples with different types of charts (bar, line, and pie) from the test dataset. These visualizations illustrate a significant correlation between deformed points and regions containing potential answers, particularly the correct answer (highlighted by the red box in Figure 5).

Model	OCR	Size	ChartQA	OpenCQA
Donut [16]	✗	201M	41.8	14.8
VisionTaPas [28]	✓	110M	45.5	-
TaPas [14]	✓	110M	41.3	-
T5 [34]	✓	220M	41.0	9.3
VL-T5 [5]	✓	220M	41.6	14.7
ChartReader [4]	✓	220M	52.6	-
Pix2Struct [18]	✗	282M	56.0	-
VL-T5$_{pre}$ [5]	✓	220M	51.8	-
VisionTaPas$_{pre}$ [28]	✓	110M	47.1	-
ChartT5 [49]	✓	220M	53.2	-
MatCha [21]	✗	300M	64.2	-
UniChart [29]	✗	201M	66.2	14.4
PaLI-17B [2]	✗	17B	47.6	-
LLaVA1.5-13B [22]	✗	13B	55.3	-
DEPLOT [20]	✗	715B	**79.3**	-
QDCHART-Donut	✗	259M	57.2	**15.6**
QDCHART-UniChart	✗	259M	**66.6**	14.6

Table 4. Comparison with baselines on ChartQA.

A common observation is that the deformed points tend to cluster around text and chart data element regions while being sparse or uniformly spaced in the blank areas. We initialize our deformable points uniformly spaced, and they move towards nearby visual elements or stay still when nothing is around. The deformed points align with the question text and visual traits; for instance, in the line graph, points cluster around the answer "2009" region, locating the answer correctly; in the bar graph, where the question asks for the "leftmost value", most points shift to the left, demonstrating the model's ability to understand and follow the question; in the pie graph, even there are two "2%" texts, the points are clustered on the one that matches the question description "refused". More examples can be found in Figure 10 in Supp. D. This demonstrates that our proposed QDCAt enhances the model's reasoning ability through the movement of deformable points. Still, QDCHART exhibits limited performance in answering questions that require mathematical reasoning. Specifically, it struggles with questions involving operations such as calculating averages, medians, products, or sums. We provide three examples of such issues in Figure 8 in Supp. D.

We evaluate our models' performance on OpenCQA using BLEU. However, BLEU has limitations in assessing semantic and factual correctness and is sensitive to generation

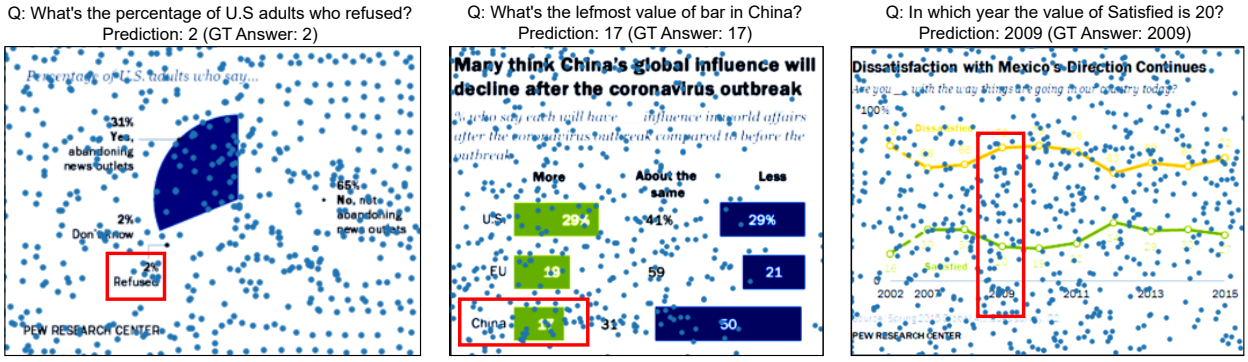

Figure 5. Qualitative examples on question-guided deformed points

parameters. To address this, we manually reviewed 50 cases and found that QDChart excels at describing visual facts, such as chart component proportions and accurately identifying values on charts. However, it struggles with more complex tasks, such as multihop reasoning and providing detailed textual explanations of charts. In addition, we provide a detailed case study in Figure 7 in Supp. B.

5.3. Ablation Studies

To demonstrate the contributions of different components to the overall performance of QDCHART, we conduct ablation studies and experiment with two additional methods to combine the features from the two image encoders.

- QDCHART - QON: We remove QON and bilinear sampling layers from QDCHART.
- QDCHART$_{\text{Concat}}$: Instead of using the deformable co-attention block, we test concatenation over the channel dimension, denoted by $\tilde{\boldsymbol{x}} \oplus \boldsymbol{y}$.
- QDCHART$_{\text{CNN}}$: We further experiment with concatenation followed by a CNN layer to adaptively fuse the spatial features, denoted by CNN$(\tilde{\boldsymbol{x}} \oplus \boldsymbol{y})$.

Model	Human	Machine	Average
QDCHART	**34.9**	**79.4**	**57.2**
QDCHART - QON	31.2	79.2	55.2
QDCHART$_{\text{Concat}}$	26.2	73.6	49.9
QDCHART$_{\text{CNN}}$	27.2	76.0	51.6

Table 5. Ablation results of QDCHART-Donut on ChartQA

The experimental results shown in Table 5 demonstrate that without our proposed deformable co-attention, using CNN fusion (QDCHART$_{\text{CNN}}$) decreases performance by 5.52%. Similarly, removing the CNN layer and using simple concatenation (QDCHART$_{\text{Concat}}$) results in a further drop in accuracy to 49.92%. Additionally, the question-guided offset network (QON) helps the model focus on visual content closely relevant to the question. When this component is removed (QDCHART - QON), accuracy decreases by 1.64%. This setting is especially influential for human-written questions and less so for machine-generated questions. Human-written questions emphasize visual and logical reasoning and pose significant challenges for previous work. We provide baseline machine and human performance in Table 7 in Supp. D. Our model demonstrates that using the question as an early multimodal fusion cue enhances the ability to answer more complex questions.

6. Conclusion

In this study, we address ChartQA by enhancing chart component recognition and proposing a novel question-aware attention fusion module. We introduce CHARTFORMER, a unified network designed to handle multiple chart comprehension tasks across various chart types in an end-to-end manner. This innovative architecture is carefully crafted to handle the intricate nuances associated with diverse chart components, offering a robust solution for accurate and reliable instance segmentation. We further propose QDCHART, which integrates a novel Question-guided Deformable Co-Attention (QDCAt) fusion block to align question-aware chart features extracted by CHARTFORMER with general-purpose multimodal encoder features. This approach explores new possibilities in multimodal fusion and enhances the guidance derived from the question. Through extensive experimentation and evaluation, our approach demonstrates exceptional performance, highlighting its effectiveness in addressing ChartQA challenges.

Acknowledgement

The authors would like to thank Mingyang Zhou for his helpful comments. This research is partially supported by the award #2238940 from the Faculty Early Career Development Program (CAREER) of the National Science Foundation (NSF) and the U.S. DARPA FoundSci Program #HR00112490370. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. VQA: Visual Question Answering, Oct. 2016. arXiv:1505.00468 [cs]. 1

[2] Xi Chen, Xiao Wang, Soravit Changpinyo, A. J. Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Hassan Akbari, Gaurav Mishra, Linting Xue, Ashish Thapliyal, James Bradbury, Weicheng Kuo, Mojtaba Seyedhosseini, Chao Jia, Burcu Karagol Ayan, Carlos Riquelme, Andreas Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. PaLI: A Jointly-Scaled Multilingual Language-Image Model, June 2023. arXiv:2209.06794 [cs]. 6, 7, 12

[3] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention Mask Transformer for Universal Image Segmentation, June 2022. arXiv:2112.01527 [cs]. 2, 4, 5, 6, 7

[4] Zhi-Qi Cheng, Qi Dai, Siyao Li, Jingdong Sun, Teruko Mitamura, and Alexander G. Hauptmann. ChartReader: A Unified Framework for Chart Derendering and Comprehension without Heuristic Rules, Apr. 2023. arXiv:2304.02173 [cs]. 1, 2, 6, 7, 12

[5] Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. Unifying Vision-and-Language Tasks via Text Generation, May 2021. arXiv:2102.02779 [cs]. 2, 6, 7, 12

[6] Jinho Choi, Sanghun Jung, Deok Gun Park, Jaegul Choo, and Niklas Elmqvist. Visualizing for the Non-Visual: Enabling the Visually Impaired to Use Visualization. *Computer Graphics Forum*, 38(3):249–260, June 2019. 2

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks, June 2017. arXiv:1703.06211 [cs]. 2

[8] Ning Ding, Ce Zhang, and Azim Eskandarian. Saliendet: A saliency-based feature enhancement algorithm for object detection for autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 9(1):2624–2635, 2024. 2

[9] Yang Ding, Jing Yu, Bang Liu, Yue Hu, Mingxin Cui, and Qi Wu. MuKEA: Multimodal Knowledge Extraction and Accumulation for Knowledge-based Visual Question Answering. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5079–5088, New Orleans, LA, USA, June 2022. IEEE. 1

[10] 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, June 2021. arXiv:2010.11929 [cs]. 2

[11] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood Attention Transformer, Apr. 2022. 3

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN, Jan. 2018. arXiv:1703.06870 [cs]. 2, 5, 6, 7

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, Dec. 2015. arXiv:1512.03385 [cs]. 2

[14] Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin 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, 2020. arXiv:2004.02349 [cs]. 2, 7, 12

[15] Shankar Kantharaj, Xuan Long Do, Rixie Tiffany Leong, Jia Qing Tan, Enamul Hoque, and Shafiq Joty. OpenCQA: Open-ended question answering with charts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11817–11837, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. 2, 6

[16] 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, Oct. 2022. arXiv:2111.15664 [cs]. 1, 2, 4, 5, 6, 7, 12

[17] Jay Lal, Aditya Mitkari, Mahesh Bhosale, and David Doermann. LineFormer: Rethinking Line Chart Data Extraction as Instance Segmentation, May 2023. arXiv:2305.01837 [cs]. 2

[18] 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, Oct. 2022. arXiv:2210.03347 [cs]. 1, 2, 6, 7, 12

[19] Chin-Yew Lin and Franz Josef Och. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 501–507, Geneva, Switzerland, aug 23–aug 27 2004. COLING. 6

[20] 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, May 2023. arXiv:2212.10505 [cs] version: 2. 3, 6, 7, 12

[21] 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, Dec. 2022. arXiv:2212.09662 [cs]. 1, 3, 6, 7, 12

[22] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved Baselines with Visual Instruction Tuning, May 2024. arXiv:2310.03744 [cs]. 6, 7, 12

[23] Xiaoyi Liu, Diego Klabjan, and Patrick NBless. Data extraction from charts via single deep neural network, 2019. 2

[24] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual Denoising Pre-training for Neu-

ral Machine Translation, Jan. 2020. arXiv:2001.08210 [cs]. 2, 4, 5

[25] 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, Aug. 2021. arXiv:2103.14030 [cs]. 2, 4

[26] Junyu Luo, Zekun Li, Jinpeng Wang, and Chin-Yew Lin. ChartOCR: Data Extraction from Charts Images via a Deep Hybrid Framework. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1916–1924, Waikoloa, HI, USA, Jan. 2021. IEEE. 2, 6

[27] Weihong Ma, Hesuo Zhang, Shuang Yan, Guangshun Yao, Yichao Huang, Hui Li, Yaqiang Wu, and Lianwen Jin. Towards an efficient framework for Data Extraction from Chart Images, May 2021. arXiv:2105.02039 [cs]. 2

[28] 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. 1, 2, 6, 7, 12

[29] Ahmed Masry, Parsa Kavehzadeh, Xuan Long Do, Enamul Hoque, and Shafiq Joty. UniChart: A Universal Vision-language Pretrained Model for Chart Comprehension and Reasoning, Oct. 2023. arXiv:2305.14761 [cs]. 1, 2, 3, 6, 7, 12

[30] Nitesh Methani, Pritha Ganguly, Mitesh M. Khapra, and Pratyush Kumar. PlotQA: Reasoning over Scientific Plots, Feb. 2020. arXiv:1909.00997 [cs]. 1, 6

[31] Shivasankaran V P, Muhammad Yusuf Hassan, and Mayank Singh. LineEX: Data Extraction from Scientific Line Charts. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6202–6210, Waikoloa, HI, USA, Jan. 2023. IEEE. 2

[32] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, 2002. 6

[33] Jingyuan Qi, Zhiyang Xu, Ying Shen, Minqian Liu, Di Jin, Qifan Wang, and Lifu Huang. The art of socratic questioning: Recursive thinking with large language models. *arXiv preprint arXiv:2305.14999*, 2023. 1

[34] 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, July 2020. arXiv:1910.10683 [cs, stat]. 2, 6, 7, 12

[35] Revant Gangi Reddy, Xilin Rui, Manling Li, Xudong Lin, Haoyang Wen, Jaemin Cho, Lifu Huang, Mohit Bansal, Avirup Sil, Shih-Fu Chang, et al. Mumuqa: Multimedia multi-hop news question answering via cross-media knowledge extraction and grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11200–11208, 2022. 1

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Jan. 2016. arXiv:1506.01497 [cs]. 2

[37] Juergen Schmidhuber. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117, Jan. 2015. arXiv:1404.7828 [cs]. 2

[38] Ying Shen, Zhiyang Xu, Qifan Wang, Yu Cheng, Wenpeng Yin, and Lifu Huang. Multimodal instruction tuning with conditional mixture of lora. *arXiv preprint arXiv:2402.15896*, 2024. 1

[39] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. SOLOv2: Dynamic and Fast Instance Segmentation, Oct. 2020. arXiv:2003.10152 [cs]. 2, 5, 6, 7

[40] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. DAT++: Spatially Dynamic Vision Transformer with Deformable Attention, Sept. 2023. arXiv:2309.01430 [cs]. 2, 3, 5, 6, 7

[41] Zhiyang Xu, Chao Feng, Rulin Shao, Trevor Ashby, Ying Shen, Di Jin, Yu Cheng, Qifan Wang, and Lifu Huang. Vision-flan: Scaling human-labeled tasks in visual instruction tuning. *arXiv preprint arXiv:2402.11690*, 2024. 1

[42] Zhiyang Xu, Ying Shen, and Lifu Huang. Multiinstruct: Improving multi-modal zero-shot learning via instruction tuning. *arXiv preprint arXiv:2212.10773*, 2022. 1

[43] Wenyuan Xue, Dapeng Chen, Baosheng Yu, Yifei Chen, Sai Zhou, and Wei Peng. ChartDETR: A Multi-shape Detection Network for Visual Chart Recognition, Aug. 2023. arXiv:2308.07743 [cs]. 2

[44] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked Attention Networks for Image Question Answering. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–29, Las Vegas, NV, USA, June 2016. IEEE. 1

[45] Ce Zhang and Azim Eskandarian. A comparative analysis of object detection algorithms in naturalistic driving videos. *ASME International Mechanical Engineering Congress and Exposition*, Volume 7B: Dynamics, Vibration, and Control:V07BT07A018, 11 2021. 2

[46] Ce Zhang, Azim Eskandarian, and Xuelai Du. Attention-based neural network for driving environment complexity perception. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2781–2787, 2021. 2

[47] Ce Zhang, Chengjie Zhang, Yiluan Guo, Lingji Chen, and Michael Happold. Motiontrack: End-to-end transformer-based multi-object tracking with lidar-camera fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 151–160, June 2023. 2

[48] Yangheng Zhao, Jun Wang, Xiaolong Li, Yue Hu, Ce Zhang, Yanfeng Wang, and Siheng Chen. Number-adaptive prototype learning for 3d point cloud semantic segmentation. In Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 695–703, Cham, 2023. Springer Nature Switzerland. 2

[49] 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 *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1314–1326, Toronto, Canada, July 2023. Association for Computational Linguistics. 1, 2, 6, 7, 12