Integrating Image Data Extraction and Table Parsing Methods for Chart Question answering

Ahmed Masry
York University
Toronto, ON, Canada
masry20@yorku.ca

Enamul Hoque Prince
York University
Toronto, ON, Canada
enamulh@yorku.ca

Abstract

Chart question answering (QA) is a challenging task, where the goal is to automatically predict the answer for a given question about a chart. Most of the existing approaches apply the visual question answering models for regular images directly on the chart images without recovering the underlying data from them. In this paper, we focus on automatically recovering the data from image chart followed by applying the state-of-the-art model for table parsing to obtain the answer to the given question. Our approach achieves impressive results on the FigureQA and Chart Question Answering Challenge (CQAC) datasets.

1. Introduction

The task of question answering with a chart has received a lot of attention today [13]. Given a chart and a natural language question, the goal is to automatically predict the answer. There are millions of charts available on the Web in bitmap image format that does not provide the underlying data and visual encodings from which those charts were built. This makes the image chart question answering task more challenging and it requires effective integration of computer vision and natural language processing.

Most existing approaches to chart question answering treat charts as regular images and often apply computer vision techniques designed for visual question answering tasks on regular images that usually contain scenes and objects (e.g. [12], [10]). However, there are two main issues with such approaches. First, chart images are very different than regular images because they visually encode the data using visual attributes (color, length) of graphical marks (e.g. rectangle, circle). Therefore, QA models that do not attempt to recover the data and visual encodings from chart may not always perform well. Second, most existing approaches treat chart QA as a classification task, where only a limited number of answers are possible (e.g. yes/no, retrieving a value represented by a graphical mark in the chart). In reality, people often ask compositional questions where answers can go beyond a limited set of possible values, as they such questions require combining multiple operations such as aggregating values, finding extremes, and calculating sums or differences of values [13].

To address the above issues, we are interested in exploring how to utilize the unique structure of charts for automatic question answering. In particular, instead of simply treating chart images as regular images, can we recover the underlying data from images first and feed them to the model to improve the accuracy? To this end, we first apply computer vision techniques to recover the data from an image chart. We then apply a question answering model for tabular data called TAPAS (stands for Table Parser) on the recovered data to generate answers [8] (as shown in Figure 1). Through experiments, we demonstrate how combining automatic chart data extraction and table parsing methods can mitigate the limitations in the current chart Question answering approaches and boost their performance.

2. Related Works

There has been growing interest in performing chart question answering [12, 10, 3, 18, 11, 13]. Several datasets and models have been proposed recently for this task. The FigureQA [12] contains 100K charts generated synthetically and around 1.3M (yes/no) Q/A pairs. Their baseline model applies relation networks [17] on the features extracted from the chart image and the question using CNN and LSTM, respectively, to infer the answer. DVQA is another synthetic Charts QA dataset which also provides template-based QA pairs that involve structural understanding, data retrieval, and resoning questions. Chaudhry et al. [3] introduced another dataset, LEAFQA, with various chart types and templates. They also proposed the LEAFNET model, an attention-based model over the encoded-question and the chart image features. Singh et al. [18] then augmented the dataset (LEAFQA++) and utilized a

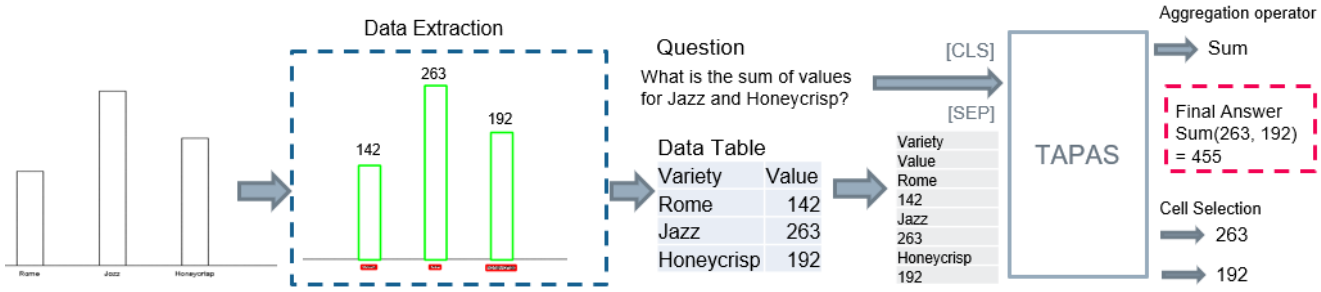

Figure 1: Our ChartQA pipeline operates in two stages: (1) data extraction and (2) table parsing using the TAPAS model

transformer-based model for this task. Kafle et al. [11] introduced the PreFIL model that fuses the learned question and image embeddings generated by CNN and LSTM efficiently to predict the answer. While the above body of work have shown promising results for chart QA, there are two issues. First, these approaches directly apply computer vision techniques on image chart without attempting to recover the underlying data [12, 10, 3, 18, 11]. Second, these models perform reasonably well when the answer comes from a fixed set of vocabulary (axis-labels, legends or common words like yes and no) while they fail to answer open-vocabulary questions that require mathematical operations (e.g. sum, average).

Works that focus on utilizing the underlying data table of chart and applying table parsing are very limited [13, 14]. Kim et al. [13] build a Chart QA dataset based on a simplified assumption that the underlying data table of a chart is already accessible to the model and directly feed that table to a table parsing method named SEMPRE [15]. PlotQA [14] also applies the SEMPRE model [15] to generate the answers but instead of assuming that the underlying data table is available it uses computer vision techniques to recover data. Our work is inspired by this line of work. However, such table parsing method (SEMPRE) needs to translate the question into a logical query as an intermediate step in retrieving the answer. Generating such logical forms can be difficult (e.g. costs for collecting logical forms and label bias problem) and answers can be limited (e.g. cannot generate 'yes'/'no' answers). To address this issue, we adopted TAPAS, the state-of-the-art model for Table QA which does not require the generation of intermediate logical form, instead it predicts the selection of relevant table cells and then applies an aggregation operator to such selection.

3. Datasets

In this work, we experiment with two datasets: (1) Figureqa [12], and (2) Chart Question Answering Challenge (CQAC) [1]. Figureqa [12] consists of synthetically plotted chart images using the Matplotlib software. The training set contains 100K charts while the validation and test sets contain 20K chart images each. There are 5 main chart types: vertical bar, horizontal bar, pie, line, and dot line charts. One million and three hundred thousand (yes/no) questions and answers pairs are generated using 15 question templates. Some templates are shown in Table 1.

Figureqa question templates	Type
Is X the maximum?	bar, pie
Is X greater than Y?	bar, pie
Is X the smoothest?	line
CQAC question templates	**Type**
Is X bigger than Y?	bar, pie
Is X bigger than the sum of Y and Z?	bar, pie
Which variety has the maximum value?	bar, pie
What is the sum of values for X and Y?	bar, pie

Table 1: Sample templates from FigureQA and CQAC.

CQAC dataset involves three tasks (low, mid, and high levels) and two chart types (bar and pie). In this paper, we are focusing on mid-level and high-level tasks. In the mid-level task, the model needs to extract the underlying data from the bar and pie chart images by measuring the height of the bars and angles of the pie sectors. The high-level task involves answering a question about a chart image. The questions are generated using 8 main templates. Some of them are shown in Table 1. These question templates can be divided into two main categories: Yes/No questions and open-vocabulary questions. The datasets has two main sets: training and testing sets. While the training set contains 160K charts (80K bar and 80K pie charts), the testing set contains 40K charts (20K bar and 20K pie charts). Since the provided test set doesn't have the labels, we randomly sample 5% of the given training set and use as a testing set (4K bar and 4K pie charts) to evaluate our approach.

4. Methodology

As illustrated in Figure 1, our QA pipeline has two main stages: (1) Data extraction from charts and (2) Table question answering. Given a chart image, we predict underlying data values, detect the text labels, and retrieve the data ta-

[1] https://cqaw.github.io/challenge

ble. Then, we feed the question and the data table as input to the TableQA model to infer the final answer.

4.1. Data Extraction

For the FigureQA dataset, we assume that the data table is given as a priori, since the dataset provides the data tables for the chart images. In the CQAC dataset, we extract the underlying data values from the chart images using the OpenCV library [2]. Since the bar charts in the CQAC dataset do not have y axis labels, the bar heights directly correspond to data values without requiring any transformation. To compute bar heights, we find rectangular contours in the image and then compute the difference between the y-coordinate of the top and bottom points of each rectangle (see 2, right and left). For pie charts, we first find the largest contour in the image. Then, we compute the center of the pie using the bounding box of the pie contour as shown in Figure 2, middle. To further split the pie into sectors, We detect the black pixels around the pie center to identify the lines separating the sectors as shown by red dots around the center in Figure 2, middle. Finally, we calculate the angles of each sector using the center point and the two sector border lines points we detected in the last stage. The final values are computed by dividing the angles by 360.

To detect the text labels in the chart images, we experimented with two models: CRAFT [1] and Google Vision API [2]. Eventually, we decided to use Google Vision since its predictions were more accurate and had less typos than CRAFT. To construct the final data tables, we associate each bar or pie sector value with its corresponding text label using distance-based heuristics.

4.2. Table QA

Tapas[8] is one of the state-of-the-art models in the Table QA literature. It is based on the BERT architechture [6] and uses additional embeddings to encode the tabular data structure such as Position ID, Segment ID, Column/Row ID and Rank ID [8]. As shown in Figure 1, the model takes the flattened data table and the question as input. Similar to BERT, TAPAS adds the `[CLS]` token at the beginning and uses the `[SEP]` token to separate between the question and the table cells. The output embeddings of the table cells tokens are fed into a classification layer to select the relevant table cells to the question. Moreover, the output of the first token, `[CLS]` is fed into another classification layer which predicts the required mathematical operation (e.g. SUM, AVERAGE, and COUNT). The operator is then applied on the selected cells and the final answer is computed.

For the CQAC high-level dataset, we split the dataset into two subsets based on the query: Yes/No questions and open-vocabulary questions. We identify Yes/No type based on presence of keywords in the question (e.g. 'is', 'does').

[2]https://cloud.google.com/vision

	Vbar	Hbar	Pie	Dot line	Line	Overall
Ours	99	99.05	98.50	91.40	91.40	96.60
PreFIL	98.80	98.09	95.11	91.82	92.19	94.84

Table 2: Results on the FigureQA dataset

Since the open-vocabulary questions are similar to WikiTQ dataset [16], we initialized our model weights using the pretrained small TAPAS model on WikiTQ and fine-tuned the model on the open-vocabulary questions using the same weakly-supervised training manner. For the yes/no questions, we omit the cell selection [7] and simply pass the `[CLS]` token output through a classification layer to predict the answer. For training, both Figureqa and CQAC (yes/no), we first initialized the Tapas model's weights with the pretrained model on the TabFact dataset [4] and then fine-tuned it on our dataset. In all experiments, we used the Adam optimizer with an initial learning rate of 0.00001.

5. Results and Discussion

As previously mentioned, we evaluate our approach on our testing split of the CQAC mid-level and high-level datasets. For the FigureQA, we compare our results with the PreFIL [11] which is the current state-of-the-art (see Table 2). Our approach achieves an overall improvement of 2% over the current state-of-the-art. We also notice that the accuracies of our model on the vertical bar, horizontal bar, and pie charts are relatively higher than the line and dot line charts. This is mainly because the questions for these chart types are simpler than the line and dot line charts questions. As shown in Table 1, the questions for vbar, hbar, and pie charts usually require applying some simple mathematical operation over two or more table cells. For example, to answer the question "Is X greater than Y?" , the model only needs to compare between two cells values. In contrast, the line and dot line charts questions require more complex operations over many table cells which makes reasoning more challenging. For example, to answer "Is X the smoothest?", the model needs to measure the smoothness of each table column cells, which is not straightforward, and compare them all to infer the correct answer. We anticipate that the large variant of TAPAS will have the computational capacity to improve the accuracies for these complex reasoning questions. Still, these results show the potential of utilizing the underlying data table by the models to improve the results on the existing chart question answering datasets.

We evaluate our data extraction approach using CQAC mid-level testing split using use three metrics: the error rate [9], accuracy, and relaxed accuracy. The mean error rate can be computed as follows:

$$Error\ rate = \sum_{i=0}^{|gt|} \frac{|gt_i - pr_i|}{gt_i}$$

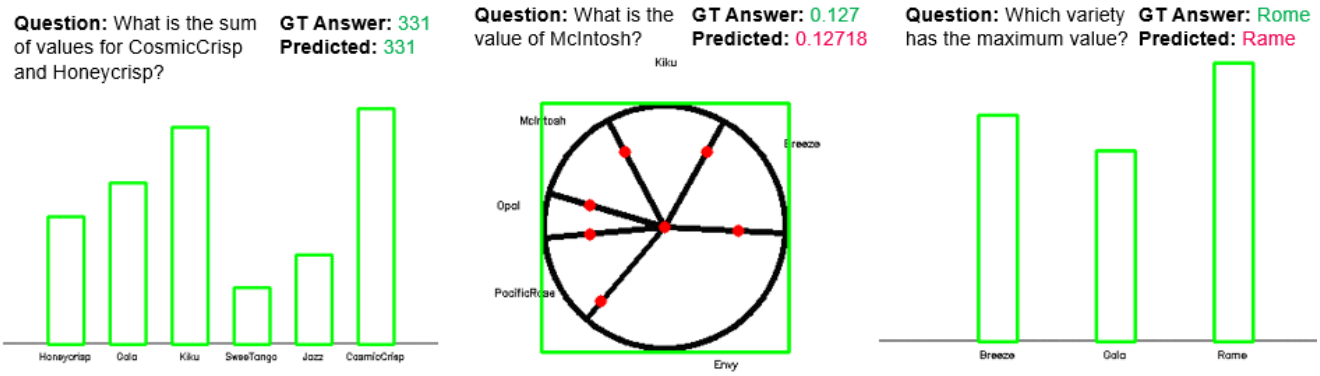

Figure 2: Some sample predictions on the CQAC dataset.

where gt_i is the ground truth value for point i and pr_i is the predicted value for point i.

The accuracy measures the number of charts whose extracted values exactly match the ground truth values. However, this accuracy measure is not ideal to evaluate chart data extraction algorithm since a small amount of noise can be introduced due to the resolution of the image, background noise etc. Consequently, we introduce the relaxed accuracy measure where a chart is considered successfull if all the data points are extracted successfully and the error rate is below a threshold (e.g. 0.02) similar to [5]. As shown in Table 3, our approach, despite its simplicity, managed to achieve 99.9% overall relaxed accuracy.

	Bar	Pie	Overall
Error rate	0.0	0.005	0.0025
Accuracy	100%	0%	50%
Relaxed Accuracy	100%	99.8%	99.9%

Table 3: Data extraction results for the CQAC mid-level set.

Question Type	Bar	Pie	Overall
Yes/No	96.83%	95.92%	96.31%
Open (Exact)	60.45%	25.27%	42.60%
Open (Relaxed Value)	60.45%	62.85%	61.38%
Open (Relaxed Value + Text)	74.78%	77.31%	75.75%
All (Exact)	75.94%	55.59%	65.56%

Table 4: Results for the CQAC high-level dataset.

Finally, we evaluate our model on the testing split of the CQAC high-level dataset. For the Yes/No questions, we only use the accuracy metric since this is a binary classification task. For the Open-Vocabulary questions, we use the accuracy and relaxed accuracy metrics, similar to above. Since the Google Vision OCR model predictions can have some typos and miss-recognized characters, the model's answer can sometimes be considered incorrect although the model selected the correct table cell (see Figure 2, right). Hence, we also add an additional metric in which we not only relax the numerical answers, but also relax the textual answers by using the levenstein-distance as a similarity metric. We consider the model's textual prediction to be correct if the levenstein distance between the predicted text and the ground truth text is less than or equal to two (e.g. Rome and Rame). We can notice that the accuracy increased from 42.60% to 61.38% by relaxing the accuracy measure over the data value (see Table 4). Since the bar charts extracted data values are identical to the ground truth values (Figure 2, left), the relaxation didn't increase the accuracy for the bar charts. However, the accuracy for the pie charts significantly increased (from 25.27% to 62.85%) since the data extraction algorithm predictions tends to deviate a little bit from the ground truth values (Figure 2, middle). Moreover, the overall accuracy further increased from 61.38% to 75.75% by relaxing the accuracy measure for the textual answers. This indicates that the Chart QA approaches are usually limited by the OCR models' performance.

To further explore the effects of bad OCR predictions we conduct another experiment where we replace the text labels in the original question with their misspelled counterparts in the extracted data tables (if the levenstein distance is at most two). As such, the modified input questions to the model become consistent with the extracted data tables. We found such modification results in improvement from 75.75% to 85.45% for the Open-Vocab (Relaxed Value+Text) setup. This again supports our hypothesis that the results are substantially affected by the OCR model's performance.

6. Conclusion and Future Work

In this paper, we combine automatic data extraction and table question answering for the Chart QA task. Our approach provides impressive results on two datasets. However, our approach needs to classify the question into two categories (open and fixed vocabulary) and train separately for each category. Moreover, our model only considers the data values of the chart image, although the charts may have other properties (such as colors, legends and axis) that are needed to be recovered. In future, we are planning to design

one unified model that can address all types of questions and recover additional chart properties.

References

[1] Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. *CoRR*, abs/1904.01941, 2019. 3

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 3

[3] R. Chaudhry, S. Shekhar, U. Gupta, P. Maneriker, P. Bansal, and A. Joshi. Leaf-qa: Locate, encode attend for figure question answering. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3501–3510, 2020. 1, 2

[4] Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*, 2020. 3

[5] J. Choi, Sanghun Jung, Deok Gun Park, J. Choo, and N. Elmqvist. Visualizing for the non-visual: Enabling the visually impaired to use visualization. *Computer Graphics Forum*, 38, 2019. 4

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. 3

[7] Julian Eisenschlos, Syrine Krichene, and Thomas Müller. Understanding tables with intermediate pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 281–296, Online, Nov. 2020. Association for Computational Linguistics. 3

[8] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online, July 2020. Association for Computational Linguistics. 1, 3

[9] Daekyoung Jung, Wonjae Kim, Hyunjoo Song, Jeong-in Hwang, Bongshin Lee, Bohyoung Kim, and Jinwook Seo. *ChartSense: Interactive Data Extraction from Chart Images*, page 6706–6717. Association for Computing Machinery, New York, NY, USA, 2017. 3

[10] Kushal Kafle, Scott Cohen, Brian L. Price, and Christopher Kanan. DVQA: understanding data visualizations via question answering. *CoRR*, abs/1801.08163, 2018. 1, 2

[11] Kushal Kafle, Robik Shrestha, Brian L. Price, Scott Cohen, and Christopher Kanan. Answering questions about data visualizations using efficient bimodal fusion. *CoRR*, abs/1908.01801, 2019. 1, 2, 3

[12] Samira Ebrahimi Kahou, Adam Atkinson, Vincent Michalski, Ákos Kádár, Adam Trischler, and Yoshua Bengio. Figureqa: An annotated figure dataset for visual reasoning. *CoRR*, abs/1710.07300, 2017. 1, 2

[13] Dae Hyun Kim, Enamul Hoque, and Maneesh Agrawala. Answering questions about charts and generating visual explanations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery. 1, 2

[14] Nitesh Methani, Pritha Ganguly, Mitesh M. Khapra, and Pratyush Kumar. Plotqa: Reasoning over scientific plots. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020. 2

[15] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China, July 2015. Association for Computational Linguistics. 2

[16] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. *CoRR*, abs/1508.00305, 2015. 3

[17] Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Timothy P. Lillicrap. A simple neural network module for relational reasoning. *CoRR*, abs/1706.01427, 2017. 1

[18] Hrituraj Singh and Sumit Shekhar. STL-CQA: Structure-based transformers with localization and encoding for chart question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3275–3284, Online, Nov. 2020. Association for Computational Linguistics. 1, 2